U.S. Department of Education
Federal Student Aid

Enterprise Testing Standards Handbook

Version 2.0

Final

September 25, 2008

Document Version Control

Document Information

Title	Enterprise Testing Standards Handbook
Revision	Version 2.0
Issue Date	September 25, 2008
Filename	ETS_Handbook.doc

Document Revision History

This section summarizes the document revision history. Each entry includes the version number of the document, the date of the change, the page number where the change occurred, and a brief description of the change. The most current change will always appear on the first row of the table.

Version	Date	Page	Description
2.0	09/17/2008	Entire Document	Update to Handbook to include new information from focus groups, industry best practices and Federal Student Aid standards.
1.0	09/28/2007	Initial Document	Initial version of Enterprise Testing Standards Handbook.

Document Audience List

Title/Roles
Primary
Federal Student Aid and/or Contractor Project Managers
Federal Student Aid and/or Contractor Test Managers/Test Leads
Federal Student Aid Enterprise Test Management
Secondary
Testing Teams
Federal Student Aid Application Owners
Application Development Teams

Table of Contents

List of Figures

List of Tables

Executive Summary

Federal Student Aid's Office of the Chief Information Officer (CIO) has developed the Enterprise Testing Standards Handbook (Handbook) as part of an initiative to standardize testing policy and practices within Federal Student Aid. The Handbook was created using feedback from Federal Student Aid focus groups, as well as industry and Federal Student Aid best practices. Policies and standards for all test phases are documented in this Handbook. The Handbook supports Federal Student Aid's efforts to achieve structure, consistency, repeatability, and continuous process improvement in software testing.

Federal Student Aid has defined a Target State Vision (TSV) to describe how it should operate and administer Title IV programs. The TSV is based upon delivering student aid in an efficient and cost-effective manner, providing the best access to customers, and maintaining appropriate levels of oversight. The Handbook is intended to provide guidance for testing of software applications developed under the TSV.

The information in this Handbook covers all phases of application testing. Details of application testing contained in this Handbook include:

- Federal Student Aid-approved standardized terminology and acronyms
- Roles and responsibilities
- Description of required test phases and test types
- Testing organization and processes
- Test planning to include such management requirements as Risk Management and Defect Management
- Required artifacts and templates
- Tools and techniques
- Standards and metrics
- Specialized testing techniques

The requirements contained in this Handbook apply to Federal Student Aid Application Owners and Contractors that provide software application testing support and management.

The Handbook will require revisions over time to reflect lessons learned, new "best practices," changes in industry methods, and the usage of new tools and techniques for software testing.

Section 1. Introduction

The Enterprise Testing Standards Handbook (Handbook) is part of an initiative to standardize testing policy and practices at Federal Student Aid. The information contained in this document is essential information for all involved in test efforts. The Enterprise Testing Group is responsible for the contents of the Handbook and will provide guidance on the information contained herein.

1.1 Purpose

This Handbook supports Federal Student Aid's efforts to achieve structure, consistency, repeatability, and continuous process improvement in software testing. It sets forth policies and standards for all aspects and phases of testing, as well as the creation of the ensuing test artifacts.

1.1.1 Scope

This Handbook contains Federal Student Aid’s testing requirements.

The focus of this Handbook is on test planning, execution of the phases of testing, and the reports and artifacts generated. These activities occur during the construction and validation, implementation, and support and improvement stages of a project. However, the testing effort must begin much earlier in the project lifecycle, during the definition stage.

The Test Manager (and Test Leads in some cases) must be involved in the project from the initial project planning, through requirements definition, and the complete system design process to obtain maximum value from the formal testing activities of each project. This is an industry best practice that results in improved planning, testability of requirements, system design, and higher quality projects.

Table 1-1, Handbook Audience

Handbook Audience	
Primary	**Secondary**
Federal Student Aid and/or Contractor Project Managers	Testing Teams
Federal Student Aid and/or Contractor Test Managers/Test Leads	Federal Student Aid Application Owners
Federal Student Aid Enterprise Test Management	Application Development Teams

All users should consult **Appendix C: Cross Reference for Intended Audience to Relevant Handbook Section** for assistance in identifying the sections of this Handbook of most importance to them.

Non-technical users should also consult **Appendix D: Testing Techniques** for an overview of fundamental testing concepts, as well as the **Glossary** in **Appendix B**.

1.2 Naming Conventions

- The terms "*application*," "*application software*," "*project*," and "*system*," used to describe the items to be tested, are defined in the *Virtual Data Center Configuration Management Database Data Dictionary*, Version 1.1, November 7, 2007.

1.3 Document Organization

The Handbook includes six narrative sections that contain the information needed to implement software testing standards and practices that meet Federal Student Aid's testing requirements.

These sections are:

Section 1. Introduction

Section 2. Selecting a Contractor – outlines the requirements for ensuring that the best contractor for the project is selected.

Section 3. Test Planning – provides for an understanding of the application, requirements, development activities and test planning documentation that will guide the testing effort.

Section 4. Testing Phases – describes application testing during development and after deployment, and includes phase descriptions and the associated standards.

Section 5. Defect Management – describes the process for recognizing, investigating, taking action on, and disposing of defects.

Section 6. Test Metrics and Reporting – outlines the efficiency and effectiveness of testing activities, test plans, and test processes throughout all phases of testing.

In addition, there are six appendices that provide supplementary information, standard document templates, and checklists to aid in implementing the standards and practices described in the narrative sections.

These appendices are:

Appendix A. Acronyms and Abbreviations – contains definitions for all of the acronyms and abbreviations used in this document.

Appendix B. Glossary – contains definitions for the major terms used in this document.

Appendix C. Cross Reference for Intended Audience to Relevant Handbook Sections – contains a cross reference for the various roles, by their associated sections.

Appendix D. Testing Techniques – contains an introduction and overview of the most common testing techniques.

Appendix E. Templates – contains the required templates that support this document and their instructions where applicable.

Appendix F. Bibliography and References – contains the guidance and references that were used to create this document.

Important information is marked by an ℹ *followed by the text.*

1.4 Handbook Terminology

Federal Student Aid recognizes that sometimes there are several terms used in the industry for the same testing concept or activity. Throughout this Handbook, there is standardized terminology related to Federal Student Aid testing to avoid confusion and ensure consistency in understanding the concepts and activities described. Federal Student Aid acquisitions will include these terms in language related to testing. Federal Student Aid and contract staff will use these terms in all testing deliverables and documentation.

1.5 Life Cycle Management (LCM) Software Testing and Release Management

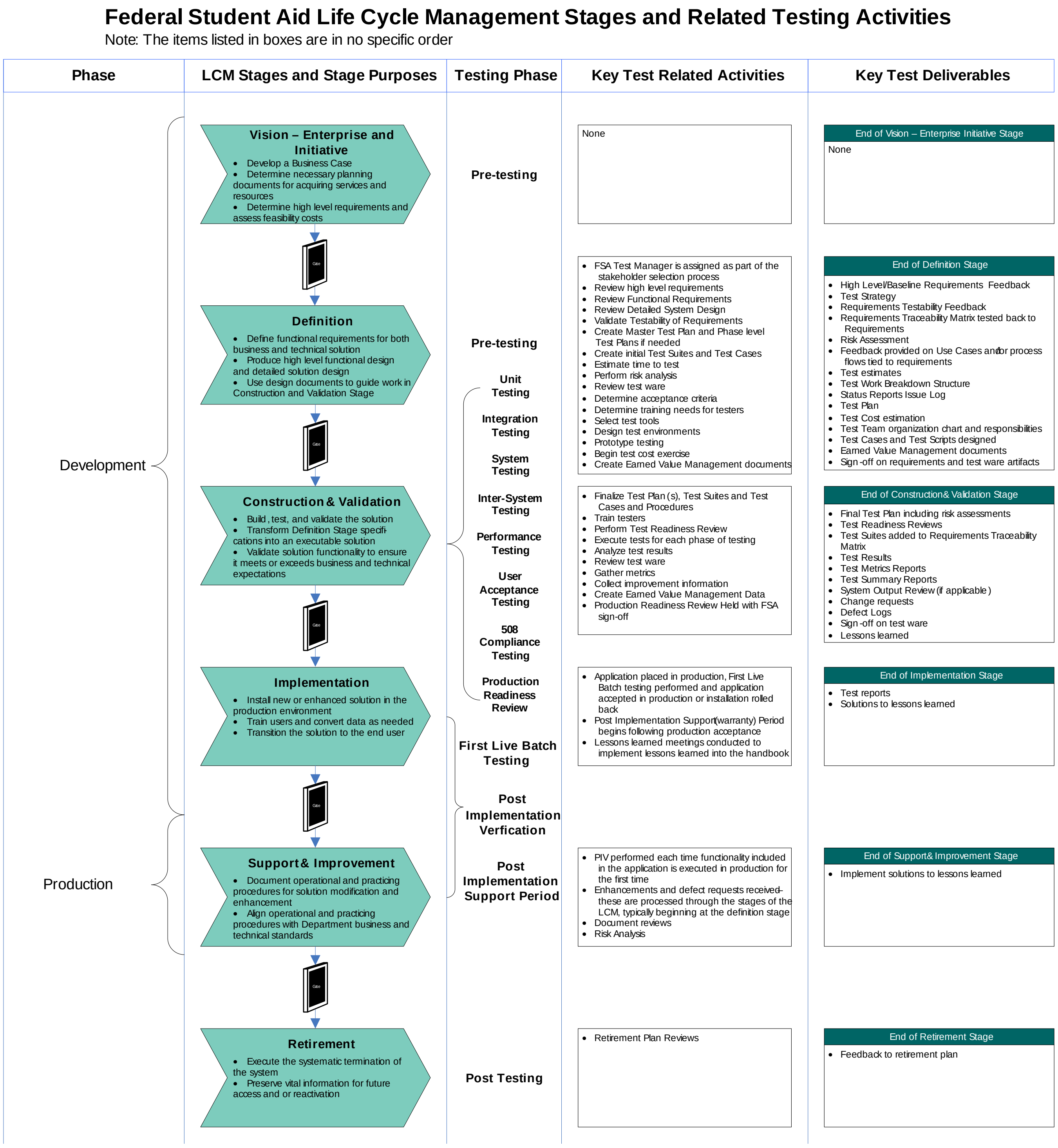
Federal Student Aid follows the U.S. Department of Education's Life Cycle Management (LCM) Framework, a structured approach for developing and delivering information technology (IT) solutions. The LCM outlines the stages required for each project, the key activities, and the core deliverables.

The Enterprise Testing Standards contained in this Handbook complement the LCM by defining specific testing phases that create periodic evaluations. These evaluations are designed to determine how well a project is meeting the requirements defined for the application.

Testing also supports release management within the LCM Framework by conducting User Acceptance Tests (UAT) prior to application deployment and First Live Batch Testing (FLB). Failure of the application during First Live Batch Testing initiates rollback of the new application and additional development and test activities to correct the defects identified during the deployment process.

Figure 1-1, Life Cycle Management Stages and Related Testing Activities, shown on the next page, describes the relationship of the Life Cycle Management System for managing applications and the related phases of testing.

Figure 1-1, Life Cycle Management Stages and Related Testing Activities



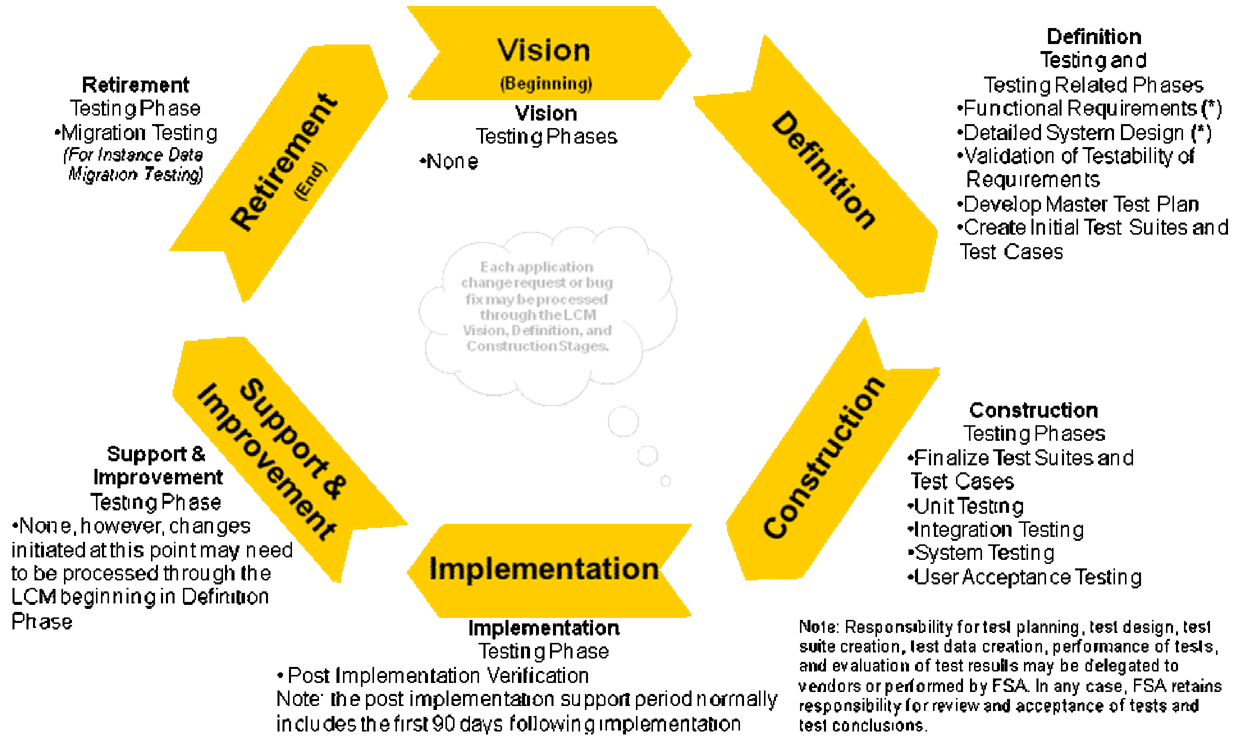
Federal Student Aid Life Cycle Management Stages and Related Testing Activities

Note: The items listed in boxes are in no specific order

Phase	LCM Stages and Stage Purposes	Testing Phase	Key Test Related Activities	Key Test Deliverables
Development	**Vision – Enterprise and Initiative** • Develop a Business Case • Determine necessary planning documents for acquiring services and resources • Determine high level requirements and assess feasibility costs	**Pre-testing**	None	**End of Vision – Enterprise Initiative Stage** None
	Definition • Define functional requirements for both business and technical solution • Produce high level functional design and detailed solution design • Use design documents to guide work in Construction and Validation Stage	**Pre-testing**	• FSA Test Manager is assigned as part of the stakeholder selection process • Review high level requirements • Review Functional Requirements • Review Detailed System Design • Validate Testability of Requirements • Create Master Test Plan and Phase level Test Plans if needed • Create initial Test Suites and Test Cases • Estimate time to test • Perform risk analysis • Review test ware • Determine acceptance criteria • Determine training needs for testers • Select test tools • Design test environments • Prototype testing • Begin test cost exercise • Create Earned Value Management documents	**End of Definition Stage** • High Level/Baseline Requirements Feedback • Test Strategy • Requirements Testability Feedback • Requirements Traceability Matrix tested back to Requirements • Risk Assessment • Feedback provided on Use Cases and/or process flows tied to requirements • Test estimates • Test Work Breakdown Structure • Status Reports Issue Log • Test Plan • Test Cost estimation • Test Team organization chart and responsibilities • Test Cases and Test Scripts designed • Earned Value Management documents • Sign-off on requirements and test ware artifacts
	Construction & Validation • Build, test, and validate the solution • Transform Definition Stage specifications into an executable solution • Validate solution functionality to ensure it meets or exceeds business and technical expectations	**Unit Testing** **Integration Testing** **System Testing** **Inter-System Testing** **Performance Testing** **User Acceptance Testing** **508 Compliance Testing** **Production Readiness Review**	• Finalize Test Plan (s), Test Suites and Test Cases and Procedures • Train testers • Perform Test Readiness Review • Execute tests for each phase of testing • Analyze test results • Review test ware • Gather metrics • Collect improvement information • Create Earned Value Management Data • Production Readiness Review Held with FSA sign-off	**End of Construction & Validation Stage** • Final Test Plan including risk assessments • Test Readiness Reviews • Test Suites added to Requirements Traceability Matrix • Test Results • Test Metrics Reports • Test Summary Reports • System Output Review (if applicable) • Change requests • Defect Logs • Sign-off on test ware • Lessons learned
	Implementation • Install new or enhanced solution in the production environment • Train users and convert data as needed • Transition the solution to the end user	**First Live Batch Testing** **Post Implementation Verfication**	• Application placed in production, First Live Batch testing performed and application accepted in production or installation rolled back • Post Implementation Support (warranty) Period begins following production acceptance • Lessons learned meetings conducted to implement lessons learned into the handbook	**End of Implementation Stage** • Test reports • Solutions to lessons learned
Production	**Support & Improvement** • Document operational and practicing procedures for solution modification and enhancement • Align operational and practicing procedures with Department business and technical standards	**Post Implementation Support Period**	• PIV performed each time functionality included in the application is executed in production for the first time • Enhancements and defect requests received – these are processed through the stages of the LCM, typically beginning at the definition stage • Document reviews • Risk Analysis	**End of Support & Improvement Stage** • Implement solutions to lessons learned
	Retirement • Execute the systematic termination of the system • Preserve vital information for future access and or reactivation	**Post Testing**	• Retirement Plan Reviews	**End of Retirement Stage** • Feedback to retirement plan

Another view of the relationship of testing to the Federal Student Aid Life Cycle Management Stages can be seen in Figure 1-2 below:

Figure 1-2, Federal Student Aid LCM Stages and Related Testing Phases


Vision
(Beginning)
Vision
Testing Phases
•None
Definition
Definition
Testing and
Testing Related Phases
•Functional Requirements (*)
•Detailed System Design (*)
•Validation of Testability of Requirements
•Develop Master Test Plan
•Create Initial Test Suites and Test Cases
Construction
Construction
Testing Phases
•Finalize Test Suites and Test Cases
•Unit Testing
•Integration Testing
•System Testing
•User Acceptance Testing
Implementation
Implementation
Testing Phase
• Post Implementation Verification
Note: the post implementation support period normally includes the first 90 days following implementation
Support & Improvement
Support & Improvement
Testing Phase
•None, however, changes initiated at this point may need to be processed through the LCM beginning in Definition Phase
Retirement
(End)
Retirement
Testing Phase
•Migration Testing
(For Instance Data Migration Testing)
Each application change request or bug fix may be processed through the LCM Vision, Definition, and Construction Stages.
Note: Responsibility for test planning, test design, test suite creation, test data creation, performance of tests, and evaluation of test results may be delegated to vendors or performed by FSA. In any case, FSA retains responsibility for review and acceptance of tests and test conclusions.

Section 2. Selecting a Contractor

2.1 Overview

Federal Student Aid relies on contractors to perform testing for most application development and maintenance projects. **Section 2.3** provides proposal evaluation guidance to help ensure that the best contractor is selected for each project.

Key members of the test team (Test Manager and Test Leads) must be certified test professionals with experience. The key members must hold at least a Certified Test Engineer (CTE) certification or equivalent credentials.

2.2 Intended Audience

Federal Student Aid Project Managers, Federal Student Aid Test Managers, and Contracting Officers will use the information in **Section 2.3** to provide guidance to contractors during the solicitation phase and as a checklist for evaluating proposals during the contractor selection process.

In addition, the Contractor should be aware of and shall perform in accordance with Federal Student Aid policies.

2.3 Proposal Checklist

Table 2-1 lists items Federal Student Aid expects to be included in the proposal from the contractor. Federal Student Aid Project Managers and Test Managers will use this guidance when selecting a testing contractor. This information will also assist Federal Student Aid when creating work statements and evaluation criteria for technical proposals.

The information provided is not exhaustive; project staff must review **Section 3. Test Planning** and **Section 4. Testing Phases** to understand the different roles and responsibilities of Federal Student Aid and contractors.

Table 2-1, Proposal Checklist

Item #	Proposal Checklist
1	Proposals must clearly state the cost of testing for the project separately from the development cost.
2	Proposal must state if the company is **Capability Maturity Model Integration (**CMMI) compliant and provide details on how the company will demonstrate their certification level.
3	Proposals must state that the Test Manager is designated as key personnel on the project. The Test Manager is expected to participate in the requirements gathering process. The Test Manager is expected to participate in the Earned Value Management baseline review training sessions and the Integrated Baseline Review.

Item #	Proposal Checklist
4	Proposals must state that all projects must have one Test Manager who may participate as a tester, but the Test Manager may not be the only tester to participate in Integration and System testing. At a minimum, there must be at least one Test Manager and one tester on every project.
5	Proposals must state the contractors will provide a Master Test Plan (MTP) and phase level test plan(s) that follow Federal Student Aid standards covering each phase of testing included in the Handbook. Contractors may add additional sections to test plans at their discretion. Each test plan is a formal deliverable.
6	Proposals must state that the contractors will provide traceability of the test suites (test cases, procedures, scripts, and scenarios) back to requirements and state the methodology and tool for this task.
7	Proposals must state the level and type of support the contractor will provide during User Acceptance Testing (UAT), when performed by Federal Student Aid.
8	Proposals must include acknowledgement that the contractor is responsible for creating the User Acceptance Test Plan and User Acceptance Test Suites with guidance from Federal Student Aid unless the contract states otherwise. The proposal must recognize that these test suites may be different or complementary to the test suites created for System Testing.
9	Proposals must include acknowledgement that the contractor must create a test summary report that complies with all sections of the template provided in the Handbook. The contractor must also provide a high-level graphical summary of test results.
10	Proposals must indicate that ample time will be allocated to review test artifacts. <u>Guidance</u> • Analyzing the test summary report to provide input into the Production Readiness Review presentation • All test related support documentation such as the Interface Control Documents must be provided to the Federal Student Aid testers for review and approval at least ten business days prior to the start of System Test • Turnaround times for reviewing artifacts may be adjusted based on complexity and size of the document and will be at the discretion of the Federal Student Aid Project Manager • The review period for the Master Test Plan will be ten business days
11	Proposals must include that requirements are approved prior to the creation of test suites (cases, procedures, scripts, scenarios) to mitigate the need to rewrite test suites.
12	Proposals must state that test suites for Federal Student Aid review and approval will be delivered at least five business days prior to the scheduled test suite review unless the contract states otherwise.
13	Proposals must indicate that testing will not begin until Federal Student Aid has approved the Test Readiness Review.
14	Proposals must specify that the contractor test manager must have at a minimum, five years of experience actively managing test efforts. Federal Student Aid project managers may require contractor test managers to have a minimum of ten years of experience based on the system

Item #	Proposal Checklist
	criticality, information sensitivity, system complexity, and cost.
15	Test teams should be comprised of certified and experienced professionals, with each test manager and test lead holding a CTE certification or equivalent as documented in the statement of work or statement of objective for the project.

Section 3. Test Planning

3.1 Overview

Federal Student Aid test planning includes understanding the application and requirements, knowing the schedule of development activities, and properly and completely creating the test-planning document(s) that will guide the testing effort. The Test Manager will use the requirements in this section and the test plan templates in **Appendix E - Templates** to create the test plans. Following these instructions ensures compliance with Federal Student Aid policies and promotes consistency and repeatability of test planning activities across all Federal Student Aid projects. For scheduling purposes, test planning is estimated to be approximately one third of the total testing effort.

The Test Manager will create a Master Test Plan (MTP) for most projects. Exceptions will be determined by the Federal Student Aid Application Owner and Project Manager based on system criticality, information sensitivity, system complexity and cost. Project size and complexity are discussed in detail in **Section 3.4.5, Test Estimation** of the Handbook. It is advisable that exceptions be discussed with the Enterprise Test Group. This plan is created before testing begins as a high-level description of the planned testing effort and may be refined throughout the project as additional information becomes available to improve the clarity and accuracy of the planned test effort. For large or complex projects, individual Phase Level Test Plans will supplement the MTP (for example, a Unit Testing Plan or a UAT Plan will be created in addition to the MTP). The MTP is finalized after approval and must not be updated when the project calls for Phase Level Test Plans.

When a MTP is required, the phase level test plans are supplemental to the MTP and these plans must go through the approval process. The MTP may be conditionally approved and later approved based on the approval of all phase level test plans.

The Test Manager will use the MTP Template to create the MTP at a high level before the project begins. The Test Manager will add additional detail as it becomes available during the project. When Phase Level Test Plans are required, the Test Manager will use the Phase Level Test Plan Template to create the Phase Level Test Plans. The MTP and Phase Level Test Plan templates can be found in **Appendix, E Templates.**

Master Test Plans are conditionally approved when phase level test plans are required.

All phase level test plans are appendices of the Master Test Plan.

The Master Test Plan is approved only after all phase level test plans are approved.

At the end of each test phase all defects must either be closed, deferred or in a state which is acceptable to the Federal Student Aid Test Manager/Test Lead.

3.2 Relevance of Test Planning to the Handbook Audience

The Test Planning section is relevant to all primary and secondary parties that make up the audience for this Handbook (see **Section 1.1.1 Scope**). The following key staff members need to fully understand and follow the requirements in the Test Planning section of this Handbook due to their associated responsibilities for this important aspect of testing:

- Federal Student Aid and/or Contractor Project Manager
- Federal Student Aid and/or Contractor Test Manager/Test Lead
- Federal Student Aid Enterprise Test Manager
- Federal Student Aid Application Owner
- Federal Student Aid and/or Contractor Application Development Team
- Federal Student Aid and/or Contractor Application Test Team

The following matrix provides staff roles and specific application testing responsibilities. Individuals that participate in test efforts will use this section as guidance for creating test plans, evaluating contractor compliance, and planning of all test related activities.

Table 3-1, Test Planning Roles & Responsibilities

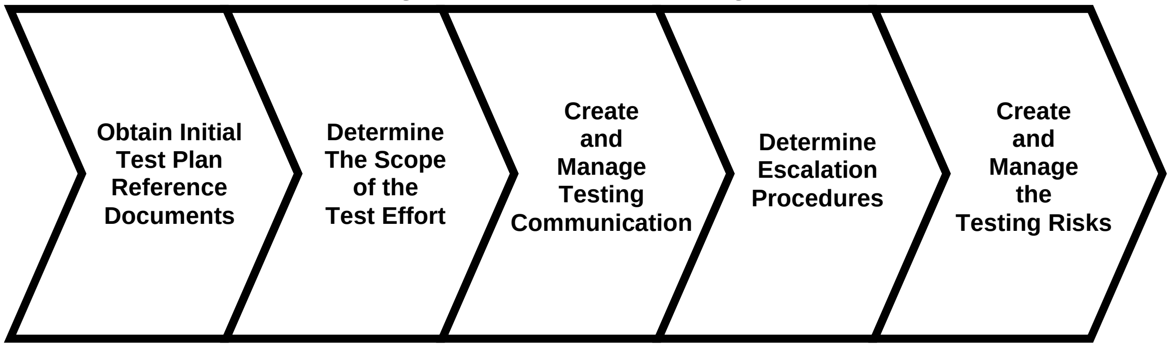
Title	Responsibilities
Project Manager	▪ Providing input to the MTP by supplying the baselined project plan document and high-level design documents to the Federal Student Aid Test Manager. ▪ Synchronizing the project plan and project schedule with the MTP. ▪ Refining the development planning. ▪ Participating in risk assessment and mitigation activities.
Test Manager	▪ Creating, maintaining, and implementing the MTP and Phase Level Test Plans (if separate from the MTP). ▪ Coordinating testing activities between the Test Team and the Project Manager. ▪ Performing ad hoc testing is appropriate in some cases because unique errors may be uncovered. When ad hoc testing is performed, the testing process must be documented step-by-step to facilitate reproducing the test during defect analysis. The Test Manager must manage this process. ▪ Participating in risk assessment and mitigation activities.
Federal Student Aid Enterprise Test Management	▪ Maintaining this Handbook to guide test planning activities (according to current Federal Student Aid policy). ▪ Evaluating contractor compliance with Federal Student Aid test planning requirements. ▪ Consults with application test managers and provides testing expertise to development projects.
Federal Student Aid Application Owners	▪ Estimating resources needed for testing by using both the information in this section and related planning information provided in individual project Test Plans.

Title	Responsibilities
	▪ Participating in testing risk assessment using information in this section and evaluating the Test Plan risk analysis provided by the Test Manager. ▪ Responsible for signing off on test artifacts.
Application Development Teams	▪ Creating and implementing the Test Plan for Unit Testing. ▪ Analyzing change requests.
Application Test Team	▪ Assist in creating and implementing the following Phase Test Plans: o Integration Test Plan o System Test Plan o User Acceptance Test Plan o User Acceptance Test Support Plan o Post Implementation Verification Plan ▪ Prioritizing test efforts and perform testing efficiently.

3.3 Test Planning Activities

The Test Manager may perform the following activities identified in **Figure 3-1, Test Planning Activities** to develop all Test Plans.

Figure 3-1, Test Planning Activities


Obtain Initial Test Plan Reference Documents
Determine The Scope of the Test Effort
Create and Manage Testing Communication
Determine Escalation Procedures
Create and Manage the Testing Risks

The following subsections provide details for each of the activities identified above:

3.3.1 Obtain Initial Test Plan Reference Documents

The Project Manager will provide the following documents to the Test Manager and Testing Team as the basis for starting the test planning process.

- **Baseline Project Plan**: This document contains appropriate delivery dates, which will be used to schedule testing activities.
- Baseline System Requirement Specifications: This document will be used to:
  - Plan the scope of testing activities
  - Plan to test both positive and negative test scenarios

- Determine which components to test
- Develop testing strategy
- Facilitate risk identification
- Determine staffing and training needs

- **High Level and Detailed Design Documents:** Information in these documents will be used to:
  - Determine test environment needs
  - Prepare Test Readiness Reviews (TRR)
  - Identify staffing and training needs
  - Create the test strategy and risk identification using the Baseline System Requirement Specifications
- **Data Management Plan:** During test planning the Data Management Plan must be reviewed in order to determine the types of testing needed, the type of conversion/balancing reports that will be tested, and to ensure that the Interface Control Documents (ICD) are available or will be created for the test effort.
- **Statement of Work Document:** This document will be used to determine the stated needs of the application owner

3.3.2 Determine the Scope of the Test Effort

Federal Student Aid will use the application system criticality, information sensitivity, system complexity and cost to determine the test plans that must be created for a project. Based on this information, Federal Student Aid will determine whether one comprehensive MTP is acceptable or if separate, detailed Phase Level Test Plans must supplement a MTP.

3.3.3 Create and Manage Testing Communication

Testing communication ensures that all key managers, the test team, and stakeholders are kept informed about testing activities, schedules, and issues. The MTP will contain a Testing Communications Plan. Since testing is critical to and affects many other areas of the application development effort as well as Federal Student Aid infrastructure, communication is vital to the success of testing projects. The Test Manager has overall responsibility for developing and managing all testing communication.

3.3.4 Determine Escalation Procedures

Test Plan(s) must include procedures and responsibilities for addressing situations when service levels are not met or testing is not successful. This process identifies specific areas where escalation to a higher level of authority is appropriate to resolve test issues. The issue resolution process contains three levels of criticality, identified as green, yellow, and red. If an issue cannot be resolved at its current level within the time specified in the Test Plan, the issue is escalated to the next level, as indicated below:

- **Green**: This is the initial default status and used when testers log issues. Testers are responsible for issues rated green
- **Yellow**: At this escalation, the Test Lead becomes involved and alternative solutions are considered
- **Red**: When escalated to red, the Test Manager becomes involved in the resolution

3.3.5 Create and Manage the Testing Risks

The principal objective of risk management is to reduce risks to an acceptable level. Risk Management, as part of the Test Plans, identifies project risks, which impact the testing effort and their associated mitigating strategies. In some cases, Risk Management for testing activities may be incorporated into the overall project risk plan. Risk management includes the following activities:

- Risk Identification
- Probability Determination
- Impact Assessment
- Mitigation Measurement Development
- Contingency Plan Development

The Project Risk Plan may contain guidance on how to define the likelihood and impact of risk. If there is no guidance in the Risk Management Plan, the MTP must state how risks will be identified and managed.

The Test Manager will assess the target system functions for criticality and risk. Criticality analysis will be based on the potential consequences associated with an error in or failure of the function. Risk assessment will also be based on the likelihood of an error in or failure of the function. The results of this analysis will be used to identify catastrophic, critical, and high-risk functions, and to focus testing resources on the most significant aspects of the system design. See Appendix D for more information on Risk Based Testing.

The definition of the acceptable level of risk and identification of those software components that are considered significant may include, for example, high-risk software using new technology or concepts. The risk evaluation activity for purposes of focusing the software testing, should be accomplished in conjunction with the formal Risk Management approach of the project. The Test Manager should work in close coordination with the Project Manager.

3.4 Detailed Test Plan Planning Activities

As additional information is identified/defined during application project planning, refinements can be made to the MTP and initial planning can begin for each phase of testing. Each Test Plan will be prepared using the appropriate Test Plan template provided in **Appendix E, Templates**. Additional sections may be required based on the type of application being tested.

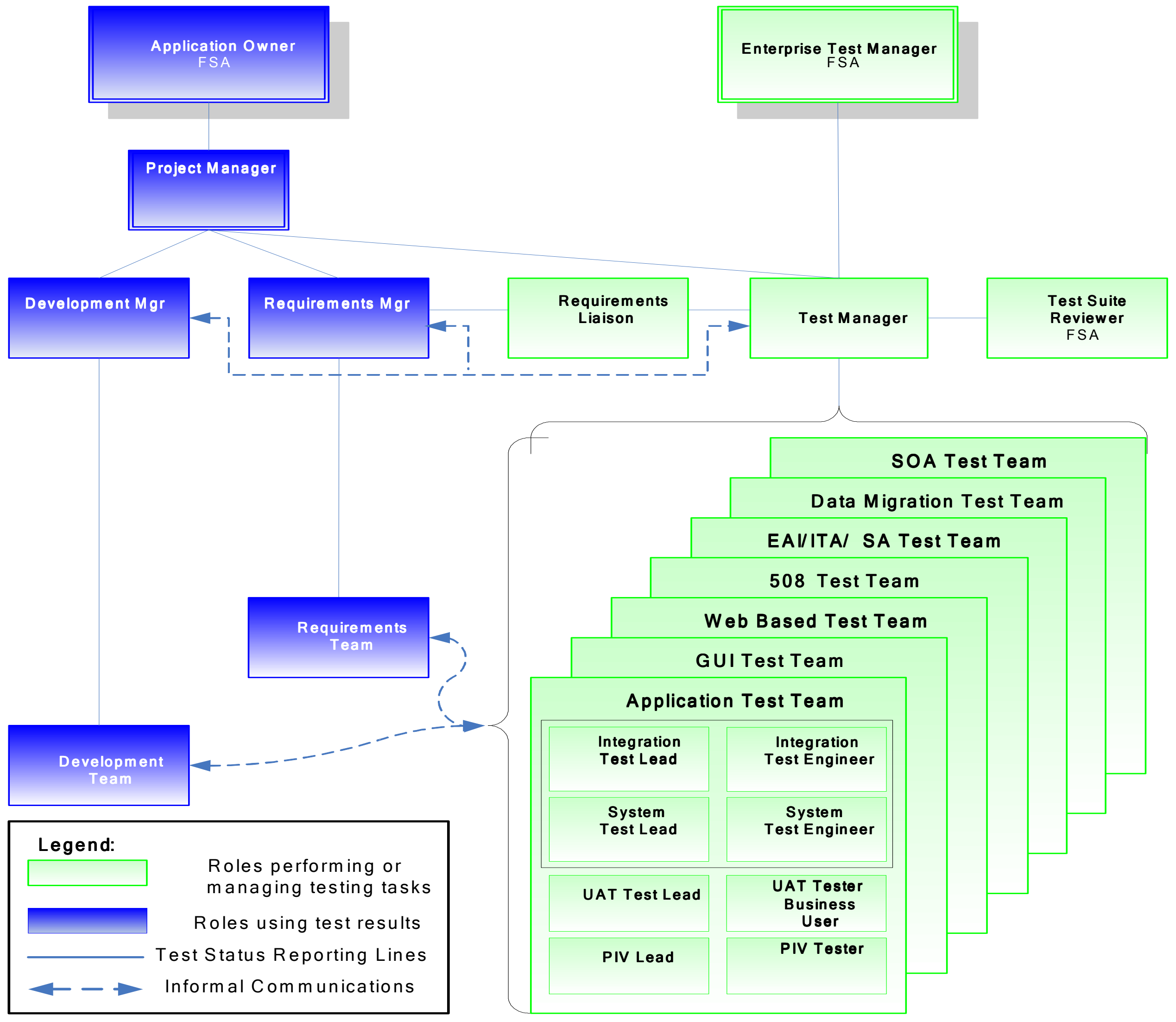
- Roles and Responsibilities
- Test Strategies
- Test Phases and Test Types
- Test Coverage
- Test Estimation
- Components To Be Tested
- Components Not To Be Tested
- Test Readiness Review
- Test Execution Cycle
- Pass/Fail Criteria
- Entry/Exit Criteria
- Suspension Criteria and Resumption Requirements
- Test Deliverables
- Environmental Needs
- Staffing and Training Needs
- Schedule
- Metrics and Reporting Planning
- Tailoring the Test Plan or Phase Level Test Plan

A description of these sections begins on page 12.

3.4.1 Roles and Responsibilities

Each Test Plan will include a section showing the roles and responsibilities of the various contractor and Federal Student Aid staff who are involved in testing activities. **Figure 3-2, Testing Related Roles and Responsibilities** illustrates the examples of roles at Federal Student Aid and their relationships.

Figure 3-2, Testing Related Roles and Responsibilities



Notes:

1. A variety of different application development test types are illustrated. Each type has all of the functions illustrated in the frame containing the Application Development Test Team.
2 . While contractor roles may vary from contract to contract, contractors normally perform development, support, and testing roles for large applications. For small applications, Development, Support, and testing are performed in-house by FSA staff.

3.4.1.1 Federal Student Aid Roles and Responsibilities

In some cases more than one person may be assigned to or participate in the activity. In these cases, the individuals work together and the most senior person has overall responsibility. The following table describes the roles and responsibilities of Federal Student Aid staff in the testing process.

Table 3-2, Federal Student Aid Roles & Responsibilities

Title	Responsibilities
Application Owner	▪ Providing resources to support testing activities. ▪ Prioritizing change requests. ▪ Participating in risk assessments.
Project Manager	▪ Determining the feasibility of performing parallel System Testing and UAT (performing testing in both phases simultaneously) due to scheduling issues. ▪ Developing appropriate risk mitigation strategies to ensure that testing is successful. ▪ Approving or rejecting recommendations to perform System Testing and UAT in parallel. ▪ Prioritizing change requests. ▪ Communicating the formal recommendation of acceptance or rejection of test deliverables to the Federal Student Aid Contracting Officer or Contracting Officer's Representative. ▪ Provides the approval to move to the next phase of testing based on the TRR.
Enterprise Test Manager	▪ Defining organization wide testing policies, standards, and ensuring compliance. ▪ Providing leadership and strategic direction to Test Managers and test teams. ▪ Developing the Enterprise Testing Schedule. ▪ Collecting Defect Data and Performing Trend Analysis. ▪ Recommending improvements related to Application Testing to the Federal Student Aid Chief Information Officer and Federal Student Aid Application Owners. ▪ Building Federal Student Aid's test information repository, Enterprise Application Testing Management Report. ▪ Maintaining the Enterprise Testing Standards Handbook. ▪ Evaluating contractor's compliance with Federal Student Aid testing requirements.

Title	Responsibilities
Test Manager	▪ Determining the feasibility of performing parallel System Testing and UAT (performing testing in both phases simultaneously) due to scheduling issues. *Note: keep in mind that parallel testing may cause risk to the project and the parallel testing decision must include a risk mitigation strategy. The final decision must include input from the Federal Student Aid Project Manager. If a contract is involved, input from the Contractor Project Manager and Contractor Test Manager must be included.* ▪ Developing appropriate risk mitigation strategies to ensure that testing is successful. ▪ Developing the recommendation and justification to perform System Testing and UAT in parallel when no contractor is involved. ▪ Developing an appropriate mitigation strategy to ensure that testing is successful. If testing problems occur, employing the above-mentioned technique. ▪ Observing the test effort performed by the contractor. *Note: the Federal Student Aid Test Manager and Project Manager will determine when the observation will take place. No formal announcement has to be given to the contractor as to when the observation will take place.* ▪ Communicating the formal recommendation of acceptance or rejection of test deliverables to the Federal Student Aid Project Manager. ▪ Prioritizing change requests. ▪ Managing the UAT process, monitoring the activity of test contractor, and participating in the TRR process; and providing a recommendation to move to the next phase of testing based on the TRR.
Development Team	▪ Designing and performing Unit Test activities ▪ Supporting subsequent testing efforts by analyzing change requests and updating code as required. ▪ Reviewing test artifacts. ▪ Reviewing and signoff of the MTP and any Phase Level Test Plans. ▪ Reviewing test reports and metrics to determine whether testing activities are on schedule. ▪ Reviewing test reports to determine that a Test Plan is being followed, and that the required artifacts are scheduled to be created. ▪ Reviewing test artifacts to evaluate the quality of the test effort. ▪ Reviewing test artifacts (for example the Test Summary Report) to evaluate contractor recommendations to move the project to the next stage of development or into production.

Title	Responsibilities
Test Lead	▪ Coordinating between the Federal Student Aid Test Manager and the Federal Student Aid Test Team. ▪ Creating, tracking, and maintaining Phase Level Test Plans in order to assure completion of testing within time, cost, and quality constraints. ▪ Identifying risks in testing and creating mitigation strategies, and taking actions in conjunction with the Test Manager and the Project Manager. ▪ Participating in analyzing and reviewing requirements for clarity, understanding, and testability. ▪ Assigning tasks and reviewing deliverables. ▪ Handling escalations from the Federal Student Aid Test Manager and Federal Student Aid Test Team. ▪ Participating in TRRs ▪ Working with the Federal Student Aid Requirements Liaison to resolve issues. ▪ Creating testing artifacts including: o Phase Level Test Plan o Phase Test Results o Phase Test Defects Reports o Phase Test Metrics o Integration Test Summary Report ▪ Communicating the formal recommendation of acceptance or rejection of test deliverables to the Federal Student Aid Project Manager. ▪ Managing UAT process, monitoring activity of test contractor, and participating in TRR process.
Testers	▪ Designing and/or providing input to, and/or maintaining Phase Level Test Suites, Test Cases and Test Scripts. ▪ Performing Peer Reviews. ▪ Reviewing the Phase Level Test Readiness Checklist. ▪ Reviewing test scripts created by contractors. ▪ Conducting phase level tests, and validating test results. ▪ Entering defects into a central defect repository and verifying their resolution. ▪ Reporting escalations to the Federal Student Aid Test Lead or Federal Student Aid Test Manager. ▪ Participating in reviewing and analyzing initiative requirements for clarity, understanding, and testability. ▪ Providing input or creating testing artifacts including: o Phase Level Test Suites, Test Cases and Test Scripts o Phase Level Test Results o Input for the Phase Level Test Defect Report
Post Implementation Verification (PIV) Testers	▪ Executing PIV Test Suites. ▪ Validating PIV Test Results. ▪ Entering defects into a central defect repository, and verifying their resolution. ▪ Reporting escalations to the Federal Student Aid PIV Team Leader. ▪ Creating testing artifacts including: o PIV Test Results o PIV Defect Report

Title	Responsibilities
Performance Test Team	*Performance testing normally occurs independently from application development. The Performance Test Team is responsible for creating the Performance Test Plan and Reports.*
Security Test Team	*Security testing may be required to meet Certification & Accreditation (C&A) for the system. If C&A is not required, the System Security Officer is required to sign off on the test plans and data being used to ensure security is being addressed.*
Requirements Liaison	▪ Understanding the application under development or modification and the related requirements and design. ▪ Facilitating communication between the project's Requirements Manager and Test Manager. ▪ Creating testing artifacts: o As determined by the Federal Student Aid Test Manager on a project-by-project basis.
Test Suite Reviewer	▪ Reviewing all test suites created by contractors. ▪ Creating testing artifacts related to identification of issues discovered. *Note: the Test Suite Reviewer may not perform hands-on testing.*
Contracting Officer	▪ Communicating the formal acceptance or rejection of test deliverables to the contractor.

3.4.1.2. Contractor Roles and Responsibilities

The following table describes the roles and responsibilities of contractor staff in the testing process.

Table 3-3, Contractor Roles & Responsibilities

Title	Responsibilities
Project Manager	▪ Determining the feasibility of performing parallel System Testing and UAT (performing testing in both phases simultaneously) due to scheduling issues. *Note: keep in mind that parallel testing may cause risk to the project and the parallel testing decision must include a risk mitigation strategy. The final decision must include input from the Federal Student Aid Test Manager, Federal Student Aid Project Manager, and Contractor Test Manager.* ▪ Synchronizing the project plan and project schedule with the MTP. ▪ Developing an appropriate risk mitigation strategy to ensure that testing is successful. ▪ Providing input to the MTP by supplying the baseline project plan document and high-level design documents to the Test Manager. ▪ Approving all the MTP and Phase Level Test Plans.
Test Manager	▪ Support UAT, which may include creating defect reports, etc. ▪ Determining the feasibility of performing parallel System Testing and UAT (performing testing in both phases simultaneously) due to scheduling issues. *Note: keep in mind that parallel testing may cause risk to the project and the parallel testing decision must include a risk mitigation*

Title	Responsibilities
	strategy. The final decision must include input from the Federal Student Aid Test Manager, Contractor Project Manager, and Federal Student Aid Project Manager. ▪ Creating, maintaining and implementing the MTP and Phase Level Test Plans (if separate from the MTP). ▪ Coordinating testing activities between the Test Team and the Project Manager. ▪ Prioritizing change requests. ▪ Developing the recommendation for using the above-mentioned technique. ▪ Participating in risk assessment and mitigation activities. ▪ Developing an appropriate mitigation strategy to ensure that testing is successful, and if testing problems occur employing the above-mentioned technique. ▪ Managing the UAT process, and participating in the TRR process.
Development Team	▪ Designing and performing Unit Test Activities. ▪ Supporting subsequent testing efforts by analyzing change requests, and updating code, as required. ▪ Providing input into the impact analysis ▪ Creating testing artifacts, including: o Unit Test Plans o Unit Test Suites and Test Scripts o Unit Test Results o Resolved Defects
Test Lead	▪ Coordinating between the Contractor Test Manager and Contractor Test Team. ▪ Creating, tracking, and maintaining the phase level test plan in order to assure completion of testing within time, cost, and quality constraints. ▪ Identifying risks in testing and creating mitigation strategies with the Contractor Test Manager and the Contractor Project Manager. ▪ Participating in analyzing and reviewing requirements for clarity, understanding, and testability. ▪ Assigning tasks and reviewing deliverables. ▪ Handling escalations from the Contractor Test Manager and Test Team. ▪ Participating in the TRRs. ▪ Working with the Federal Student Aid Requirements Liaison to resolve issues. ▪ Creating testing artifacts, including: o Phase Level Test Plan o Phase Test Results o Phase Test Defect Reports o Phase Test Metrics o Phase Test Summary Report ▪ Managing the UAT process, and participating in the TRR process.

Title	Responsibilities
Testers	▪ Designing and maintaining Phase Level Test Suites, Test Cases and Test Scripts. ▪ Performing Peer Reviews. ▪ Reviewing the Phase Level Test Readiness Checklist. ▪ Conducting phase level tests, and validating test results. ▪ Entering defects into a central defect repository and verifying the resolution. ▪ Reporting escalations to the Contractor Test Lead or Contractor Test Manager. ▪ Participating in reviewing and analyzing initiative requirements for clarity, understanding, and testability. ▪ Creating testing artifacts including: o Phase Level Test Suites, Test Cases and Test Scripts o Phase Level Test Results o Input for the Phase Level Test Defect Report
Post Implementation Verification Testers	▪ Executing PIV Test Suites. ▪ Validating Post PIV Test Results ▪ Entering defects into a central defect repository, and verifying the resolution. ▪ Reporting escalations to the contract PIV Test Team Leader. ▪ Testing artifacts created by contract PIV Testers including: o PIV Test Results o PIV Defect Report
Performance Test Team	Performance Testing normally occurs independently from application development. The Performance Test Team is responsible for creating the Performance Test Plan and Reports.
Security Test Team	Security testing may be required to meet Certification & Accreditation (C&A) for the system. If the system is not required to perform C&A, the System Security Officer is required to sign off on production data that may be used during testing and should sign off on test plans to ensure security is being addressed.

3.4.2 Test Strategies

Test strategies involve the approach and methods that will be used to plan, conduct, and report testing activities. Organizational goals and objectives and/or specific application project goals and objectives may influence the selection of test strategies. The test strategy section of the plan will specifically address the following organizational test strategies and project test strategies:

Table 3-4, Test Strategies

Organization Test Strategy	Project Test Strategy
▪ Standardization of all software testing activities across projects ▪ Building a strong testing organization ▪ Evaluating the current staffing model ▪ Setting up the service level agreements	▪ Test coverage ▪ Test tools ▪ Training requirements ▪ Types of metrics used ▪ Configuration Management process

Organization Test Strategy	Project Test Strategy
▪ Timely review of existing processes ▪ Test automation effort in terms of percentage of tests automated ▪ Handling of meetings and other organizational processes	▪ Level of Regression Testing

3.4.3 Test Phases and Test Types

There are five testing phases that occur during application development and deployment. An important part of Test Planning is to identify test types that will be performed during each test phase. **Table 3-5** lists standard test types and corresponding testing phases in which the testing is either required or recommended:

Table 3-5, Standard Test Types

	Testing Phase				
Standard Test Types	Unit	Integration	System	UAT	PIV
Component Testing	▲	-	-	-	-
Integration Testing	●	-	-	-	-
Component Interface Testing	-	▲	-	-	-
Subsystem Testing	-	●	▲	▲	●
Functional Testing	-	-	▲	▲	▲
Intersystem Testing	-	-	●	▲	●
Regression Testing	-	-	▲	▲	-
Performance Testing	-	-	●	-	-
Usability Testing	-	-	●	▲	●

Legend: ▲: Required, ●: Recommended

In addition to the standard test types shown in **Table 3-5,** depending on the nature of the project, there are specialized tests that may be required. The Test Team must consider the need for each of these specialized test types when preparing test plans. A discussion of specialized testing can be found in **Appendix D: Testing Techniques**.

Table 3-6, Specialized Test Types

Specialized Test Types	Testing Phase				
	Unit	Integration	System	UAT	PIV
508 Compliance Testing	●	●	▲	▲	-
Commercial off the Shelf (COTS) Testing	●	●	▲	▲	-
Documentation Testing	●	-	●	●	-
Evergreen Testing	-	-	▲	▲	●
Graphical User Interface (GUI) Testing	-	-	▲	●	●
Client Server Architecture Testing	●	●	●	●	-
Service Oriented Architecture Testing	▲	▲	▲	▲	▲
Web Based Architecture Testing	▲	▲	▲	▲	-

Legend: ▲: Required, ●: Recommended

In all testing phases and types, the approach should be to "break" the system (negative testing) in addition to demonstrating that the functionality works.

3.4.4 Test Coverage

Test coverage is a quality measure used in software testing. It describes the degree to which the source code of a program has been tested. To measure how well the program is exercised by a test suite, one or more *coverage criteria* are used. There are a number of coverage criteria, the main ones being:

- **Function coverage –** has each function in the program been executed?
- **Statement coverage –** has each line of the source code been executed?
- **Condition coverage** (also known as Branch coverage) – has each evaluation point (such as a true/false decision) been executed?
- **Path coverage –** has every possible route through a given part of the code been executed?
- **Entry/Exit coverage –** has every possible call and return of the function been executed?

Selection of a Test Coverage Tool to determine the test coverage provided is a test planning activity. The Test Manager, in conjunction with the Project Manager, may decide to capture test coverage to:

- Identify the portions of the application that tests have not exercised

- Accumulate coverage data over multiple runs and multiple builds
- Merge data from different programs sharing common source code
- Work with the unit testing tool to make sure that errors are found throughout the application
- Generate test coverage reports
    - These reports include, at a minimum, code location information, statement coverage and condition coverage

Before delivering a build to the Testing Team, the Development Team will configure the test coverage tool for the build to be tested. During execution, the Project Manager is responsible for ensuring the creation of periodic test coverage reports using the test coverage tool and providing them to the test team.

3.4.5 Test Estimation

Test estimation provides inputs for resource and schedule planning. The Test Manager will estimate the test effort using multiple variables, constants, and calculations. The following example describes an acceptable estimation methodology.

This method uses five variables as follows:

Table 3-7, Test Estimation Values

Variable	Description
Complexity Constant	The complexity constant provides a means of defining the additional effort required to perform moderately complex and complex tests. Using a factor of one for relatively simple tasks, a multiplier is used to increase the value of the complexity constant for more complex tasks. In the example below, a moderately complex task is estimated to require 20% more effort than a simple task (1+(1*20%)) =1.2 and a complex task is estimated to require 50% more effort than a simple task (1+(1*50%)) =1.5. Initially, these multipliers are pure estimates. As actual data is accumulated and analyzed, and as tools and processes evolve, these values should be reevaluated to determine whether the values should be adjusted to provide more accurate estimates.
Number of Requirements	The number of requirements represents the number of tasks that need to be completed. For this method to produce reasonable estimates, the definition of a requirement must be consistent from phase to phase within a project, as well as consistent from project to project.
Basic Hours	Basic Hours is an estimate based on experience that is used to calculate the time required to perform a task.
Regression Ratio	The regression ratio is equal to the number of defects divided by the number of test cases. This value is initially estimated. As testing results are reported, the estimate is adjusted based on

Variable	Description
	experience.
Regression Runs	Regression runs is the number of times a test was performed.

For example:

Step 1 – Determine Complexity Constant Values based on whether the complexity of the requirement to be tested is Low, Medium or High.

Table 3-8, Complexity Constant Values

Complexity Constant Value	Description	Complexity Constant
Low	Relatively Simple	1
Medium	Moderately Complex	1.2
High	Complex	1.5

Step 2 – Apply the Complexity Constant Value to Calculate the Subtotal Hours.

Subtotal Hours = Number of Requirements * Complexity Constant * Basic Hours

Table 3-9, Calculate Estimated Subtotal to Complete Testing

Number of Requirements	Complexity Constant	[1]Basic Hours	[2]Sub-Total
A	B	C	A*B*C=D
30	1.5	2	90
10	1.2	2	24
30	1	2	60

Step 3 – Calculate the estimated time to complete testing.

Total Hours = Subtotal Hours * Regression Ratio * Regression Runs

Table 3-10, Calculate Estimated Time to Complete Testing

[2]Sub-Total	[3]Regression Ratio	[4]Regression Runs	[5]Estimated Hours
D	E	F	D*E*F=G
90	0.5	3	135 Hrs
24	0.5	3	36 Hrs
60	0.5	3	90 Hrs
		Total Estimated Hours	261 Hrs

If testing activities are performed by the contractor, the contractor's proposal must state how test estimates are calculated.

3.4.6 Components To Be Tested

Test Plans must include the list of components (functions or requirements) that will be tested or demonstrated by the test. The component list in the plan will include a high-level description of each component without providing technical details.

Traceability to requirements and design is required in the test plan to identify the test design specification associated with each feature and each combination of features. Federal Student Aid follows the **Institute of Electrical and Electronics Engineers (IEEE) Standard (Std) 829-1998,** which instructs Test Managers to "specify the minimum degree of comprehensiveness desired. Identify the techniques that will be used to judge the comprehensiveness of the testing effort (e.g., determining which statements have been executed at least once). Specify any additional completion criteria (e.g., error frequency). The techniques to be used to trace requirements should be specified."

To ensure that all requirements are tested, the Requirements Traceability Matrix (RTM) must be completed and delivered to Federal Student Aid prior to the start of testing. Test Suites should include requirements and provide full traceability from the test suite and test case to the requirements being tested at the step level. Developers and testers should be working from the same requirements documents. In addition, this documentation should be base-lined and under configuration management control.

3.4.7 Components Not To Be Tested

This section of the Test Plan describes the list of components (functions or requirements) that will not be tested. The component list in the plan will include a high-level description of each component without providing technical details. Federal Student Aid follows IEEE Std 829-1998, which includes the requirement to identify all features and significant combinations of features that will not be tested and the reasons.

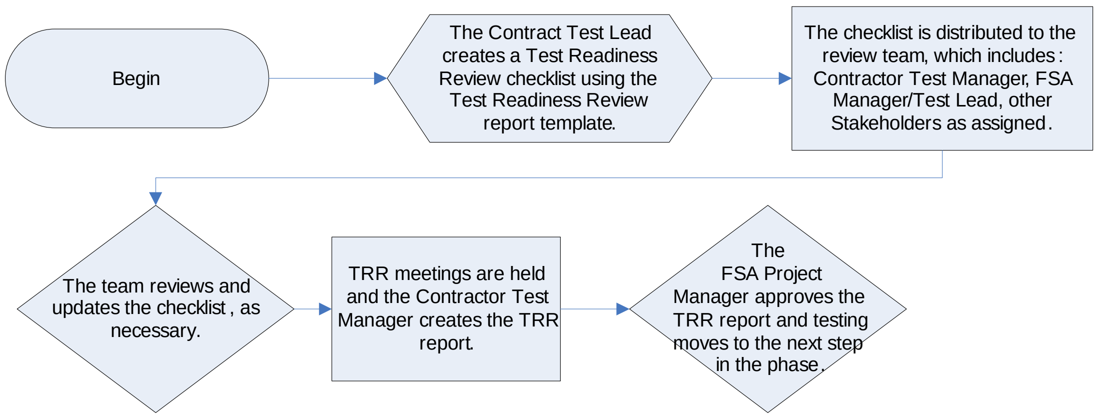
3.4.8 Test Readiness Review

The Test Readiness Review (TRR) process is the first step in Integration Testing, System Testing, UAT, and Post Implementation Verification. In each case, the TRR functions as the phase gate signifying that the requirements for beginning the next phase of testing conditions have been met. The TRR must include communication of defects found in previous test phases to the test team involved in the next phase of testing.

Testing by Contractor

When contractors perform testing, the steps in the TRR process are as follows:

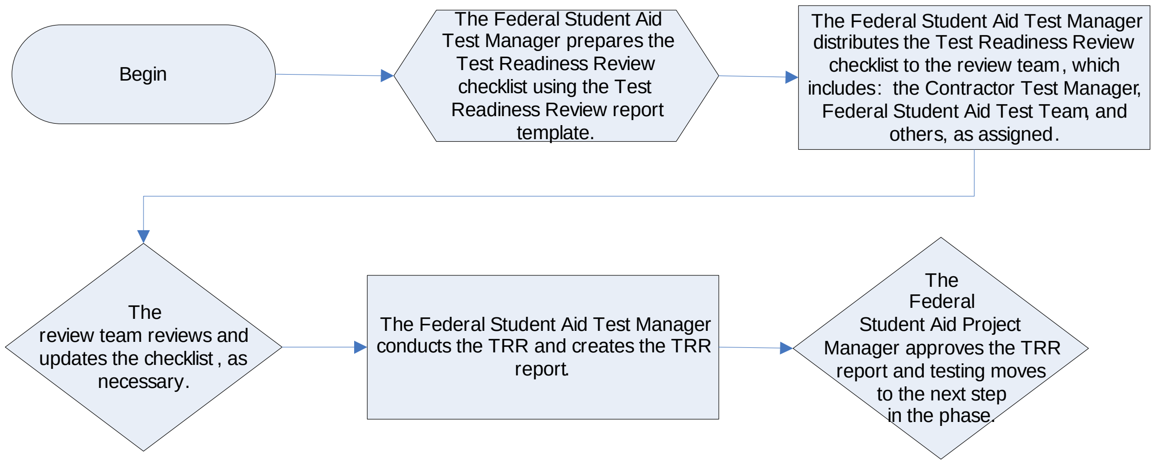
Figure 3-3, Testing by Contractor


Begin
The Contract Test Lead creates a Test Readiness Review checklist using the Test Readiness Review report template.
The checklist is distributed to the review team, which includes: Contractor Test Manager, FSA Manager/Test Lead, other Stakeholders as assigned.
The team reviews and updates the checklist, as necessary.
TRR meetings are held and the Contractor Test Manager creates the TRR report.
The FSA Project Manager approves the TRR report and testing moves to the next step in the phase.

Testing by Federal Student Aid

When Federal Student Aid performs testing for smaller projects, the steps in the TRR process are as follows:

Figure 3-4, Testing by Federal Student Aid


Begin
The Federal Student Aid Test Manager prepares the Test Readiness Review checklist using the Test Readiness Review report template.
The Federal Student Aid Test Manager distributes the Test Readiness Review checklist to the review team, which includes: the Contractor Test Manager, Federal Student Aid Test Team, and others, as assigned.
The review team reviews and updates the checklist, as necessary.
The Federal Student Aid Test Manager conducts the TRR and creates the TRR report.
The Federal Student Aid Project Manager approves the TRR report and testing moves to the next step in the phase.

3.4.9 Test Execution Cycle

Each planned test execution is a Test Execution Cycle. The number of Test Execution Cycles may change depending on the defects encountered in the system during the test execution.

The test execution cycle involves the following steps:

- Creation of a test suite which contains the test scenarios, test cases, test procedures, and test scripts
- Execution of all the test cases in the test suite
- Reporting test results
- Defect management
- Implementation of a Test Suite
  - Can typically begin as soon as a test case is defined.
  - An approved software build is usually required before executing a Test Suite to capture measurable test results.

3.4.10 Pass/Fail Criteria

Test success is based on pass/fail criteria. Federal Student Aid requires that results be clearly documented in the Test Suites. If the actual result is the same as the expected result the test passes; otherwise it fails. This section of the Test Plan contains the specific criteria that will be used for Pass/Fail assessment. In addition, dry runs of testing should be held by the test team prior to formal reviews with Federal Student Aid to validate the expected results. The plan will include the following three steps related to Pass/Fail criteria:

1. Establish precondition for the test.
2. Execute the test.
3. Verify the test result.

Expected results must be clearly documented in the Test Suites and the results should be verified during testing.

3.4.11 Entrance/Exit Criteria

Test Plan contains the criteria that determine when testing within any given test phase begins (entrance criteria) and ends (exit criteria). Entrance and exit criteria must be specified for all test efforts.

3.4.12 Suspension Criteria and Resumption Requirements

In some cases, testing must be stopped based on certain conditions or situations. For example, if an application is under test and the login feature does not work, the tester will suspend testing activity and resume once the application has been fixed. This section of the Test Plan describes all the conditions that may result in the suspension and the resumption of the testing activity.

3.4.13 Test Deliverables

This section of the Test Plan contains a list of artifacts, such as status reports and charts, required by the contract as formal test deliverables.

3.4.14 Environmental Needs

This section of the Test Plan includes the hardware details, test data details, simulators and other support required to perform testing.

3.4.15 Staffing and Training Needs

This section of the Test Plan includes the number of resources and skills required for the testing project and the identification of areas where training is required.

3.4.16 Schedule

This section of the Test Plan contains the testing schedule in the form of a work breakdown structure.

3.4.17 Metrics and Reporting Planning

The metrics that will be used to determine the success of testing activities must be included in the Test Plan. Additional information related to test metrics and reporting can be found in **Section 6 Test Metrics and Reporting**. **Subsection 6.4 Metrics Identification and Selection** may be especially useful.

3.4.18 Tailoring the MTP or Phase Level Test Plan

When it becomes necessary to tailor a Test Plan that deviates from the direction in this Handbook, the Test Manager will provide the following information and submit it to the Federal Student Aid Test Manager for approval:

- Item number (for tracking each requested change)
- Applicable Handbook Section/Subsection number and title
- Requirement as stated in the Handbook
- Tailored proposal
- Rationale for tailoring

3.5 Test Plan Reviews, Revisions, and Acceptance

Test Plan(s) are living documents that require frequent review and revision. Changes affecting the Test Plan may also affect major documents (such as test scenarios and scripts) associated with testing activities. All revisions of any lifecycle document must adhere to the required versioning control or change management to ensure that only the latest versions of documents are implemented in any lifecycle phase.

3.5.1 Test Plan Reviews

The Test Manager is responsible for reviewing and updating the MTP and Phase Level Test Plans (when required) for all other phases of testing prior to and during each test phase to ensure the most up-to-date project-related information is included and considered from a testing perspective. The Federal Student Aid Project Manager must work with the Contractor to ensure that the planned Test Plan Reviews are line items in the overall project schedule. The review process includes:

- Distributing the updated Test Plan to the Federal Student Aid Test Manager for review and comment
    - The Federal Student Aid Project Manager will work with the Federal Student Aid Test Manager to assign a team to perform the review
    - The Federal Student Aid Test Manager will recommend acceptance of the test plan
- Conducting a meeting with the assigned review team to discuss the input and identify changes.
- Incorporating changes and distributing the updated plan to the review team.
- Conducting additional reviews and revisions as necessary, until the review team agrees the revised Test Plan meets Federal Student Aid’s needs.

3.5.2 Test Plan Revisions

There will be numerous times during the test project where revisions to Test Plans and to other documents (i.e., test suites, test scenarios, test cases, test scripts, and test procedures) may be required. Reasons for revisions may include:

- Receiving new information
- Changes in the environment
- Development-testing actions
- Changes in schedules
- Addition of stakeholders

The Project Manager is responsible for ensuring that revisions are made promptly and that all relevant information associated with the changes is captured.

1. The Project Manager will provide revised Test Plans and other test project related documents to the Federal Student Aid Project Manager and Federal Student Aid Test Manager within the period required by the contract, or as directed by the Federal Student Aid Project Manager or Federal Student Aid Test Manager.
2. The Federal Student Aid Project Manager or Federal Student Aid Test Manager will review each revision for content and accuracy.

3.5.3 Test Plan Approvals

The revision process is as follows:

1. The Federal Student Aid Project Manager or Federal Student Aid Test Manager will approve the revised Test Plan(s) and related documents, or return them for correction.
2. Acceptance of the initial and each revised Test Plan and major test document occurs when Federal Student Aid signs the acceptance template. At a minimum, the Federal Student Aid Project Manager and Federal Student Aid Test Manager must sign the acceptance document.
3. The Federal Student Aid Project Manager informs the Federal Student Aid Contracting Officer as to the recommendation of acceptance or rejection of the test plan.

Section 4. Testing Phases

4.1 Overview

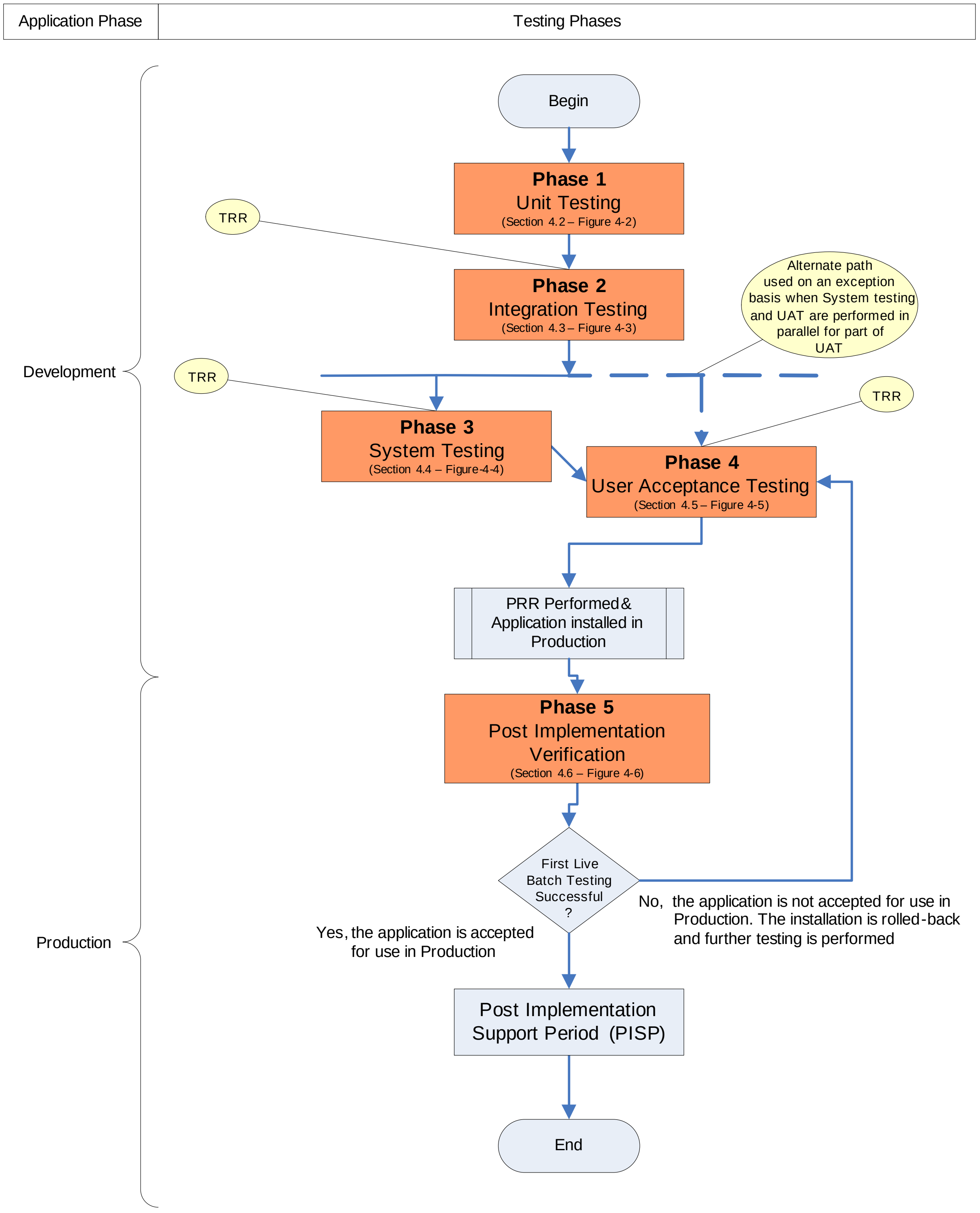
This section describes application testing during development and after deployment and includes a description of the testing phases and the standards associated with each one. There are five testing phases required by Federal Student Aid, four during application development, and one following release of the application to production.

The processes, requirements, and artifacts included in **Section 4** are the minimum required to meet Federal Student Aid Standards. Additional procedures, requirements, and artifacts may be required to ensure compliance with the intent of this Handbook.

Figure 4-1, Overview of Testing Phases on the following page shows the overall testing process across each of the five Federal Student Aid required test phases.

This figure also indicates the processes Federal Student Aid will use for evaluating the readiness of an application for production and when it occurs relative to the individual testing phases. These evaluation processes include the Production Readiness Review (PRR) and First Live Batch Testing.

Figure 4-1, Overview of Testing Phases


Application Phase
Testing Phases
Development
Production
Begin
Phase 1
Unit Testing
(Section 4.2 – Figure 4-2)
TRR
Phase 2
Integration Testing
(Section 4.3 – Figure 4-3)
Alternate path used on an exception basis when System testing and UAT are performed in parallel for part of UAT
TRR
Phase 3
System Testing
(Section 4.4 – Figure-4-4)
TRR
Phase 4
User Acceptance Testing
(Section 4.5 – Figure 4-5)
PRR Performed & Application installed in Production
Phase 5
Post Implementation Verification
(Section 4.6 – Figure 4-6)
First Live Batch Testing Successful ?
No, the application is not accepted for use in Production. The installation is rolled-back and further testing is performed
Yes, the application is accepted for use in Production
Post Implementation Support Period (PISP)
End

4.1.1 Application Development Testing

During application development, four testing phases occur:

- Phase 1: Unit Testing
- Phase 2: Integration Testing
- Phase 3: System Testing
- Phase 4: User Acceptance Testing

Testing in these phases is required for new applications as well as maintenance. In addition, all four of the test phases apply to Commercial-off-the-Shelf (COTS) software, COTS software with modifications, and custom developed applications. The four test phases occur sequentially, with the possible exception of System Testing and User Acceptance Testing (UAT). With supporting documentation and Federal Student Aid Project Manager and Test Manager approval, these two phases may occur with some overlap.

The tests performed and the related artifacts produced during these testing phases enable Federal Student Aid to evaluate new and upgraded software to determine whether the software is ready for use in a production environment.

Following UAT, key Test Team members will participate in the PRR to verify that the application is ready for the production environment.

Testers will comply with the Privacy Act of 1974 as amended, 5 U.S.C § 552a, whenever test data is obtained from a data set containing personal information.

The Project and Test Managers will evaluate the applicability of each phase to each project. If a phase is not applicable, the test manager will document why it does not apply in the MTP. The Federal Student Aid Project and Test Managers will approve the approach.

4.1.2 Post Implementation Verification (PIV)

Phase 5, Post Implementation Verification is the final test phase and occurs once the application is moved to production. Two specific types of testing may occur during this phase:

- First Live Batch Testing
- Post Implementation Support Period Testing

4.1.3 Organization

Sections 4.2 through **4.6** are organized by phases, from Phase 1 to Phase 5. Each phase contains subsections with the following information:

- Description of the testing that occurs
- Table of roles, responsibilities and artifacts
- Process diagram

- Entrance Criteria that must be completed before the test phase starts
- Test preparation guidance
- Test execution information
- Test reporting requirements
- Exit Criteria that must be met before the phase is completed and further testing can begin

4.2 Phase 1: Unit Testing

During Unit Testing, developers design and perform tests to ensure that the code written performs according to the application's design. At the successful conclusion of Unit Testing, the Test Team recommends advancing the code to the next phase of development – Integration Testing. There are two types of Unit Testing:

- **Component Testing**: Component Testing is the testing of individual components of the code. A component is the smallest product of code that accomplishes a specific action.
    - Component Testing is always required as part of Unit Testing
- **Unit Integration Testing**: Unit Integration Testing occurs when developers combine components so that several individual components are tested together.
    - Unit Integration Testing is required if integrated components are created during development

Federal Student Aid requires that Unit Testing be performed using selected path testing within the code module, as well as code coverage tools. The Federal Student Aid Project Manager will agree on the percent of the affected branch conditions and lines of code developers must test for each project. The development team is responsible for developing test drivers and stubs, as needed, to perform these tests. **Appendix D** of the Handbook discusses useful tools, including code coverage tools.

Example - Unit Testing

Unit Testing Scenario

The "last name" field on the Free Application for Federal Student Aid (FAFSA) application module may contain a maximum of 30 characters.

Test

The development tester validates that the "last name" field on the FAFSA application only allows 30 characters. The developer was able to enter 31 characters as a last name on the FAFSA application. The developer looks at the internal code to determine why more than 30 characters were allowed in the field and makes a correction to the code. The developer then retests the scenario to ensure that the "last name" field only accepts 30 characters.

The Federal Student Aid Project Manager and Federal Student Aid Test Manager must be invited to all meetings related to Unit Testing and should have insight into all Unit-Testing Activities.

Close interaction between the Project Manager and the Development Team is critical for Unit Testing to be successful and for the code to be advanced to the next testing phase. **Table 4-1, Unit Testing Roles, Responsibilities, and Artifacts,** shows the Project Manager and Development Team roles and responsibilities as required by Federal Student Aid and the artifacts that must be created for successful Unit Testing.

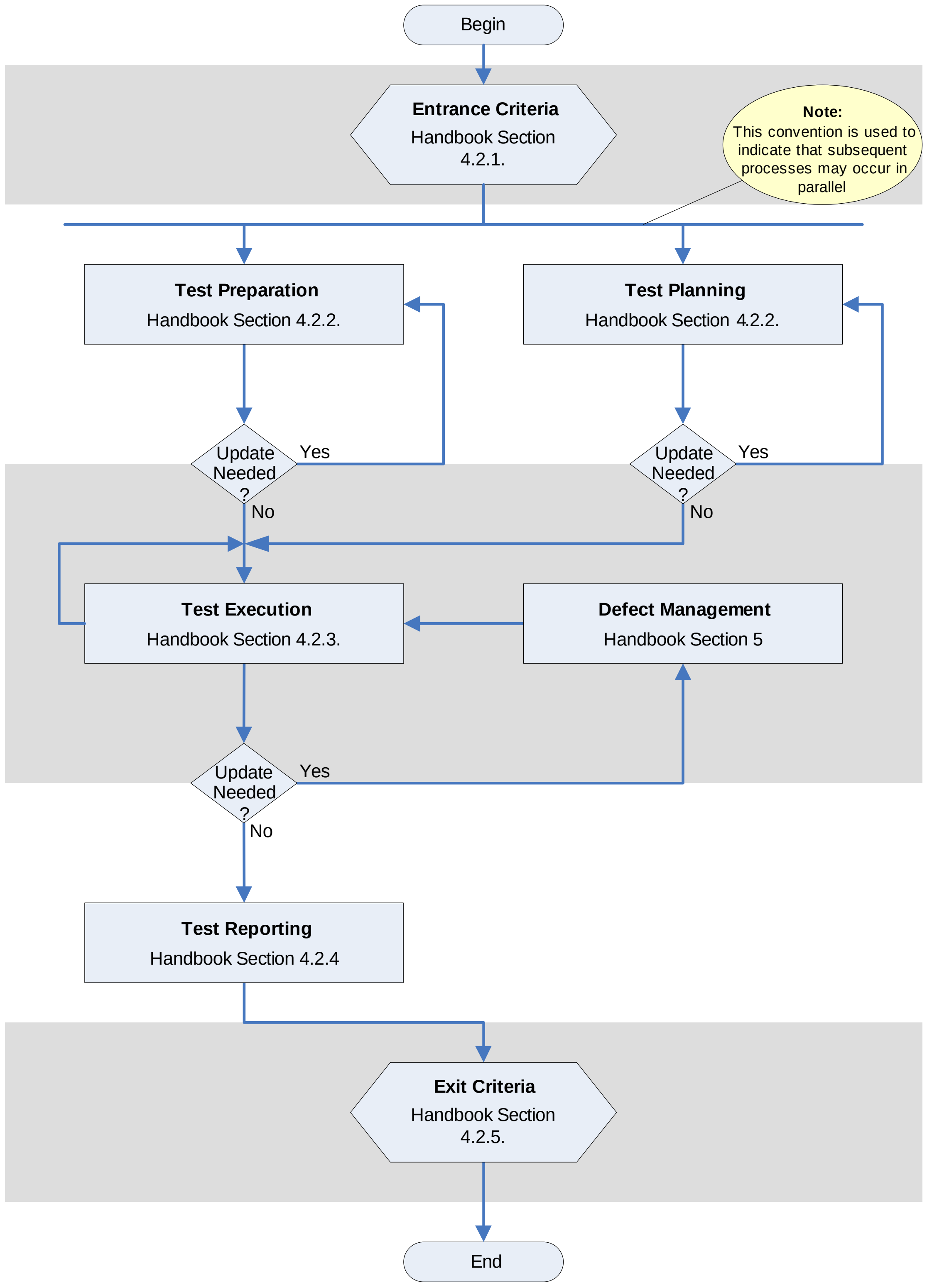
Table 4-1, Unit Testing Roles, Responsibilities, and Artifacts

Role	Responsibility
Project Manager	Plans for testing and testing related activities
	Ensures that all Entrance Criteria are satisfied prior to beginning Test Preparation. Minimum Entrance Criteria include having: ▪ Baseline MTP ▪ Baseline Detailed Design Document ▪ Approved Requirements Traceability Matrix ▪ Approved Unit Test Plan ▪ Unit Test Environment ▪ Coded software components that will be tested ▪ Completed a review of change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase
	Oversees testing activities
	Reviews all testing artifacts
	Ensures that all Exit Criteria are satisfied prior to submitting Unit Testing Artifacts to the Federal Student Aid Project Manager. Minimum Exit Criteria include ensuring that: ▪ Planned Unit Test Cases pass ▪ Urgent and high priority defects identified during Unit Testing are resolved ▪ Artifacts, reports, and metrics are updated by the developer to reflect the current state of the components tested ▪ The Project Manager has reviewed and approved all artifacts ▪ The Federal Student Aid Deliverable Review Process is completed
Federal Student Aid Test Manager	When a contractor is used, the Federal Student Aid Test Manager has the option of reviewing the Unit Test results but does not have the authority to approve them.
Federal Student Aid Architect	The Federal Student Aid Architect may review unit test results and may request the development team to perform additional testing.
Development Team	Includes the Federal Student Aid Project Manager in all project correspondence, meeting invitations, other communications and reports.

Role	Responsibility
	Prepares for Unit Testing by ▪ Updating the Unit Test Plan (as needed) ▪ Creating and updating the Unit Test Framework (where required - e.g., stubs and drivers) ▪ Creating and updating Unit Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform Component Testing and (if applicable) Integrated Component Testing ▪ Creating and updating Unit Test Data that: o Is reusable when appropriate o Is generated utilizing an automated tool whenever possible o Includes identifying and planning for additional resources if test data is to be provided from a source outside the development group
	Executes Unit Testing by ▪ Executing Unit Test Cases using the Unit Test framework ▪ Evaluating Unit Test results ▪ Performing Defect Management to verify, log, track and retest defects ▪ Updating code to resolve defects
	Reports Unit Testing by ▪ Updating Unit Testing Test Suites ▪ Updating Unit Testing Plans ▪ Creating Unit Testing Reports ▪ Creating Unit Testing Metrics ▪ Creating the Unit Testing Defect Report ▪ Submitting all artifacts to the Project Manager for review and approval

The following figure, **Process Overview of Unit Testing** shows the Unit Testing process.

Figure 4-2, Process Overview of Unit Testing



4.2.1 Entrance Criteria

Before Unit Testing can begin, Entrance Criteria must be met. Federal Student Aid may define additional entrance criteria for a project above the Federal Student Aid minimum requirement. A contractor may also recommend extra criteria for Federal Student Aid approval; however, the Federal Student Aid Project Manager must approve any exceptions to the minimum criteria. Entrance criteria conditions include at a minimum:

- Baseline MTP
- Baseline Detailed Design Document
- Approved Requirements Traceability Matrix
- Approved Unit Test Plan
- Unit Test Environment
- Coded software components that will be tested
- Completion of a review of change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase

4.2.2 Test Preparation

Along with the Entrance Criteria, the Development Team must prepare for testing by:

- Updating the Unit Test Plan (as needed)
- Creating and updating the Unit Test Frameworks (where required - e.g. stubs and drivers)
- Creating and updating Unit Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform Component Testing and (if applicable) Unit Integration testing
- Identifying and planning for additional resources if test data is to be provided from a source outside the development group
- Creating and updating Unit Test Data that:
  - Is reusable when appropriate
  - Is generated utilizing an automated tool whenever possible

4.2.3 Test Execution

Once the test preparations are complete, the Development Team performs Unit Testing by:

- Executing Unit Test Cases using the Unit Test frameworks
- Evaluating Unit Test results
- Performing Defect Management to verify, log, track and retest defects
- Updating code to resolve defects

Note: *Useful testing techniques during Unit Testing include, but are not limited to, White Box Testing, Code Review and Walkthrough, API Testing, Automated Testing, Regression Testing, and Security Testing. The* ***Glossary in Appendix B*** *provides descriptions of each technique.*

4.2.4 Test Reporting

The Development Team prepares the following series of reports and submits them to the Project Manager for review and approval. Additional reports may be required for a specific project.

- Updated Unit Test Suites
- Updated Unit Test Plans
- Unit Test Reports
- Unit Test Metrics
- Unit Test Defect Report

4.2.5 Exit Criteria

Exit Criteria define the quality standards that qualify the code for promotion to the next level or development stage. The following conditions are the minimum Exit Criteria for Unit Testing cases. (Federal Student Aid and/or the testing contractor may define additional Exit Criteria for a project – Exceptions must be approved by the Federal Student Aid Project Manager.)

- All Unit Test Cases pass
- All urgent and high priority defects identified during Unit Testing are resolved
- All artifacts (including test suites and automated test scripts), reports, and metrics are updated by the development team to reflect the current state of the components tested
- The Project Manager has reviewed and approved all artifacts
- The Federal Student Aid Deliverable Review Process was applied to each artifact
- When a contractor is used, the Federal Student Aid Test Manager and the System Architect may review the contractor's test results and may require additional testing

4.3 Phase 2: Integration Testing

Once Unit Testing is completed, the integration team integrates software units and code modules created by the Development Team with each other to create higher-level system components until all units are integrated, creating a complete system (application).
Integration testing will be required for complex new development. The Federal Student Aid project manager must determine when Integration testing is required based on the criticality, information sensitivity, complexity, and cost of a system.
Integration Testing Test Types include:

- **Component Interface Testing:** Tests the interfaces between code modules and software units
  - Integration Testing continues until developers have integrated all of the code to create the final application

- Federal Student Aid requires integration testing for applications

- **Subsystem Testing**: Required if Subsystems are created during development

As integrated components are created, Testers can begin Component Interface Testing, testing the interfaces between code modules. Depending upon how the design requirement is defined, Integration Testing continues until developers have integrated all of the code to create and test the final application.

In large applications, these components can be integrated to form subsystems that are integrated with other subsystems to create the completed application.

Integration Testing is an iterative process occurring with components integrated again and again until the application is completed. Once the first Integration Testing cycle is complete, the Test Manager will modify the Integration Test Plan and resource estimates as needed for subsequent Integration Testing Cycles. To facilitate defect resolution, testers should have an open communication path with developers prior to and during Integration Testing.

Example - Integration Testing

Integration Testing Scenario
The FAFSA website has a feature to allow the user to perform a search of schools by state.
Test
The integration tester enters the state code “VA” in the "Federal School Code” search field. The tester validates that the FAFSA website and the Federal School Code search feature work together to return only Virginia schools as a result on the FAFSA web page.

The Integration Test Team, including the Test Manager and Integration Testers, plans the testing activities for the interfaces to be tested. The Integration Test Plan includes the assigned tasks, schedule, and resources. Planning must include review of the Unit Component Test Defect Report and the Unit Integration Test Defect Report to identify problems and areas of concern that occurred during Unit Testing and that may need additional scrutiny during Integration Testing.

Table 4-2, Integration Testing Roles, Responsibilities, and Artifacts, shows the Project Manager, Test Manager, and Integration Test Team roles and responsibilities as required by Federal Student Aid, and the artifacts that must be created for successful Integration Testing. Test Suites that have been approved by Federal Student Aid may be changed during test execution, but must be reported to the Federal Student Aid test manager indicating the reason for the change. The Federal Student Aid project team will review the updated test suites to ensure the test meets the requirements. When analysis of defects leads to requirement clarification or modification, then new Test Suites must be created and approved by Federal Student Aid.

Table 4-2, Integration Testing Roles, Responsibilities, and Artifacts

Role	Responsibility	Artifact(s)
Project Manager	Provides information for testing and testing related activities	▪ Updated Project Development Plan ▪ Updated Work Breakdown Structure
	Provides TRR approval	

Role	Responsibility	Artifact(s)
Federal Student Aid Test Manager	Completion of the Federal Student Aid Deliverable Review Process	Federal Student Aid Test Manager sign-off after reviewing and approving artifacts
	If a contractor is used, the Federal Student Aid Test Manager is responsible for the following: ▪ Reviewing and approving test artifacts ▪ Monitoring testing activities ▪ Provides TRR recommendation	Contractually related recommendation to accept or reject test artifacts.
Test Manager	Ensures all Entrance Criteria are satisfied prior to beginning Test Preparation. Minimum Entrance Criteria include having: ▪ Baseline MTP ▪ MTP ▪ Detailed Design Document ▪ Approved Requirements Traceability Matrix ▪ Approved Integration Test Plan ▪ Integration Test Environment ▪ Coded Software components that will be tested ▪ Integrated code modules completed by Developers and ready to be tested ▪ TRR documentation and process ▪ Tests assigned to testers ▪ Test Suites, Test Cases, Test Scripts, and Test Procedures created ▪ Completed a review of change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase ▪ An Application login and password	▪ Baselined MTP ▪ Completed TRR ▪ Approved Integration Test Plan ▪ Integration Test Environment exists
	Oversees testing activities	▪ Email ▪ Meeting Minutes ▪ Weekly Status Reports
	Reviews all testing artifacts	▪ Reviewed Integration Test Cases ▪ Reviewed Integration Test Results

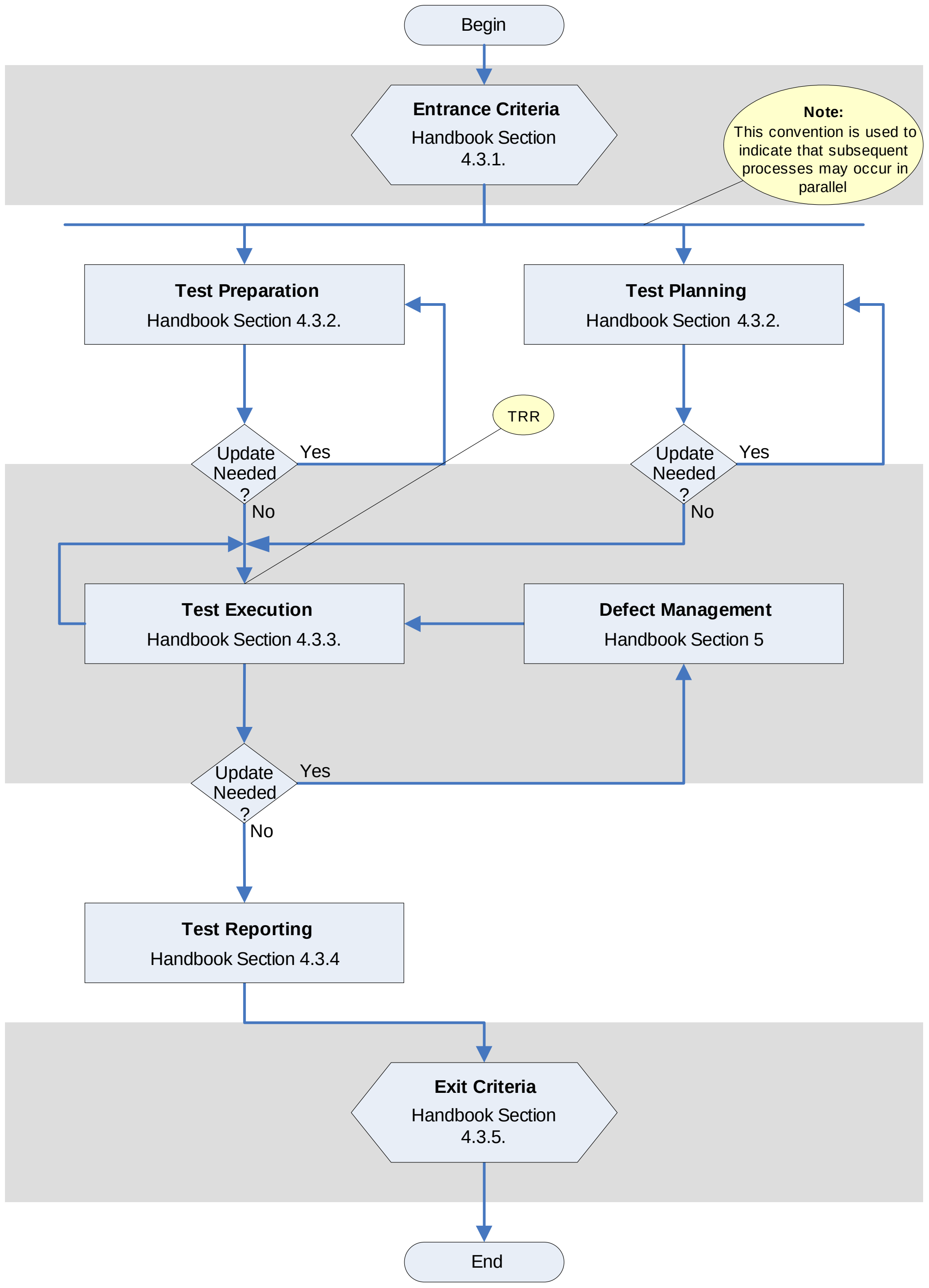
Role	Responsibility	Artifact(s)
	Ensures that all Exit Criteria are satisfied prior to submitting Integration Testing Artifacts to the Federal Student Aid Project Manager. Minimum Exit Criteria include ensuring that: ▪ All planned Integration Test Cases pass ▪ All urgent and high priority defects identified during Integration Testing are resolved ▪ All artifacts, reports, and metrics are updated by the Test Team to reflect the current state of the components tested ▪ The Test Manager has reviewed and approved all artifacts	▪ Completed Integration Test Cases ▪ Defect Log ▪ Updated artifacts, reports, and metrics ▪ Test Manager sign-off after reviewing and approving artifacts
Integration Test Team	Prepares Integration Testing by: ▪ Updating the Integration Test Plan (as needed) ▪ Determining and obtaining the Test Environment ▪ Determining and obtaining the Integration Test Framework required to execute Integration Test Cases ▪ Creating and updating Integration Test Suites, including determining test scenarios, test scripts, and test procedures to perform Component Integration Testing, and, if applicable, Subsystem Testing ▪ Obtaining logins and passwords to interface(s) ▪ Generating Integration Test data that: o Is reusable when appropriate o Is generated utilizing an automated tool whenever possible o Is generated from data that has been processed on the system and that can be re-used for testing	▪ Updated Integration Test Plan (if needed) ▪ Integration Test Framework (as needed, e.g., stubs and drivers) ▪ Integration Test Suite (including test scenarios, test cases, test scripts, and test procedures) ▪ Integration Test Data

Role	Responsibility	Artifact(s)
	Executes Integration Testing by ▪ Executing Integration Test Cases using the Integration Test Framework ▪ Evaluating Integration Test Results ▪ Performing Defect Management to verify, log and retest defects	▪ Executed Integration Test Cases ▪ Evaluated Integration Test results ▪ Complete documentation of defects in the defect management tool ▪ Updated code that resolves defects ▪ Documentation to show that defects were tracked and managed to the resolution
	Reports Integration Testing by ▪ Updating Integration Testing Test Suites ▪ Updating Integration Testing Plans ▪ Creating Integration Testing Reports ▪ Creating Integration Testing Metrics ▪ Creating the Integration Testing Defect Report ▪ Submitting all artifacts to the Project Manager for review and approval	▪ Updated Integration Test Suites ▪ Updated Integration Test Plan ▪ Integration Test Summary Report ▪ Integration Test Metrics ▪ Integration Test Defect Report

The contractors responsible for Integration Testing will ensure that tests are performed on all components that make up the integrated components of the application. At each step of the integration process, the test must verify the functionality and reliability of the application, according to approved design specifications.

Figure 4-3, Process Overview of Integration Testing, is a process flow diagram that introduces the high-level activities within Integration Testing. Additional detail for each activity is located in this section.

Figure 4-3, Process Overview of Integration Testing



4.3.1 Entrance Criteria

The following conditions are the minimum Entrance Criteria that must exist before Component and Subsystem Integration Testing can begin. (Federal Student Aid and/or the Testing Contractor may define additional Entrance Criteria for a project – Exceptions must be approved by the Federal Student Aid Project Manager.)

- Baseline MTP
- Baseline Detailed Design Document
- Approved Requirements Traceability Matrix
- Unit Testing of components is complete and all urgent and high defects found have been addressed and resolved
- Integrated code modules have been created and are ready to be tested
- Integration Test Environment has been created
- Integration Test Plan has been created
- Test Readiness Review has been completed
- Tests have been assigned to Testers
- All Test Suites, Test Cases, Test Scripts, and Test Procedures have been created
- All change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase, have been reviewed
- An Application login and password have been assigned
- Other criteria may be required based on the needs of the project

4.3.2 Test Preparation

The Integration Testing Team is responsible for performing the following tasks before and, as needed, during Component Integration and Subsystem Testing:

- Updating the Integration Test Plan (as needed)
    - When applicable, the Integration Test Plan must include communication guidelines between each subsystem. The Test Manager or Project Manager of the application under test is responsible for working with the Test Manager and Project Manager of the other subsystem to coordinate this testing phase.
- Determining and obtaining the Test Environment
- Determining and obtaining the Integration Test Framework required to execute Integration Test Suites
- Creating and updating Integration Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform Component Integration Testing and, if applicable, Subsystem Testing
- Obtaining logins and passwords to interfaces

- Generating Integration Test data that:
  - Is reusable when appropriate
  - Is generated utilizing an automated tool whenever possible
  - Is generated from data that has been processed on the system and that can be re-used for testing

4.3.3 Test Execution

Integration Test execution tasks performed by the Integration Test Team include:

- Executing Integration Test Cases using the Integration Test Framework
- Evaluating Integration Test Results
- Performing Defect Management to verify, log, and retest defects

Note: *Testing techniques useful during Integration Testing include, but are not limited to, White Box Testing, Code Review and Walkthrough, Regression Testing, Communication Testing, and Security Testing. The Glossary in Appendix B provides descriptions of each technique.*

4.3.4 Test Reporting

Integration Test reporting includes, but is not limited to, the following:

- Updated Integration Testing Test Suites
- Updated Integration Testing Plans
- Integration Testing Reports
- Integration Testing Metrics
- Integration Testing Defect Report

4.3.5 Exit Criteria

Exit Criteria define quality standards that qualify the code for promotion to the next level or development stage. The following conditions are the minimum Exit Criteria for Integration Testing cases. (Federal Student Aid and/or the Testing Contractor may define additional Exit Criteria for a project – Exceptions must be approved by the Federal Student Aid Project Manager.)

- All Integration Test Cases have been executed
- All urgent and high priority defects identified during Integration Testing are resolved
- All the artifacts (including test cases and automated test scripts), reports, and metrics are updated by the testing team to reflect the current state of the interface(s)
- The Test Manager has reviewed and approved all artifacts
- The Federal Student Aid Deliverable Review Process is completed

- Test coverage reported to Federal Student Aid
- When a contractor is used, the Federal Student Aid Test Team reviews the contractor's artifacts. The Federal Student Aid Test Manager recommends acceptance or rejection to the Federal Student Aid Project Manager and Contracting Officer and may require the contractor to perform additional testing.
- Other criteria may be required based on the needs of the project

4.4 Phase 3: System Testing

Following Integration Testing, Testers perform System Testing. System Testing evaluates the integrated system (application) as a whole. The Testing Team performs tests to ensure that each function of the system works as expected and that any errors are documented, analyzed, and resolved appropriately. The structure of the contracts will determine how the testing procedures in this section are applied to each contractor.

System Testing Types include:

- Functional Testing (Required)
- Regression Testing (Required)
- Intersystem Testing (Required if applicable)
- Performance Testing (Required if applicable)

Specialized Testing Types applicable during System Testing include:

- 508 Compliance Testing (Required if applicable)
- Security Testing (Required if applicable)

Regression Testing is performed during System Testing to ensure that changes made to the application do not adversely affect existing functionality. (*Note: Regression Testing can also be performed to ensure that changes to other systems have not had an adverse effect on the system under test.*) Performance Testing is dependent on the satisfactory completion of all other planned System Test types. The Entrance Criteria, Test Preparation, Test Execution, Test Reporting, and Exit Criteria for Performance Testing are contained in **Section 4.4.6**.

The System Testing Team, including the Test Manager and System Testers, plans System Testing activities. The Test Manager will plan the types of tests that the Test Team will perform. To the extent that these test types are not interdependent for a particular application, tests may be performed sequentially, in any order, or if staffing permits, in parallel. The System Test Plan includes the assigned tasks, schedule, and resources. Planning must include review of the Unit Testing Defect Reports and Integration Testing Defect Reports to identify problems and areas of concern that occurred in previous testing phases and that may need additional scrutiny during System Testing. When System Testing includes Intersystem Testing, the Project and Test Managers are responsible for coordinating testing activities with the major players of subsystems that may be affected by the testing activities.

End to end test scenarios must use the same data from the beginning to the end of the test. Federal Student Aid requires verification points to be contained within the same test suite. For instance, if an end to end test requires data entry, database updates and creation of reports, all three tests must be performed using the same input starting with data entry, reviewing of the database updates, and analyzing the reports to ensure that the data is correct.

Example - System Testing

System Testing Scenario

The FAFSA website has a feature to allow the user to perform a search of schools by state.

Test

The system tester enters the state code “VX” in the "Federal School Code" search field. The tester validates that the FAFSA website returns an appropriate error message since “VX” is an invalid state code.

Table 4-3, System Testing Roles, Responsibilities, and Artifacts, summarizes Federal Student Aid’s minimum requirements for the roles, responsibilities, and artifacts related to System Testing.

Test Suites that have been approved by Federal Student Aid may be changed during test execution after notification is given to Federal Student Aid as to the reason for the change. When analysis of defects leads to requirement clarification or modification, then new Test Suites must be created and approved by Federal Student Aid.

Table 4-3, System Testing Roles, Responsibilities, and Artifacts

Role	Responsibility	Artifact(s)
Project Manager	Provides information for testing and testing related activities	▪ Updated Project Development Plan ▪ Updated Work Breakdown Structure
Federal Student Aid Project Manager	If a contract is involved, work with the Federal Student Aid Test Manager, and Contractor Project Manager, and Contractor Test Manager, to complete the following: ▪ Evaluate any need to perform parallel System Testing and UAT (performing testing in both phases concurrently) ▪ Develop an appropriate mitigation strategy to ensure that testing is successful ▪ Keep track of test efforts to ensure that the schedule is being met ▪ Communicate recommendation of acceptance or rejection of the test artifacts to the contracting officer	▪ Approved parallel testing plan and related mitigation strategy

Role	Responsibility	Artifact(s)
Contract Project Manager	Works with the Federal Student Aid Project Manager, Federal Student Aid Test Manager, and Contractor Test Manager to evaluate the need to perform parallel System Testing and UAT (performing testing in both phases simultaneously) for part of each testing phase and to develop an appropriate mitigation strategy to ensure that testing is successful.	▪ Approved parallel testing plan and related mitigation strategy

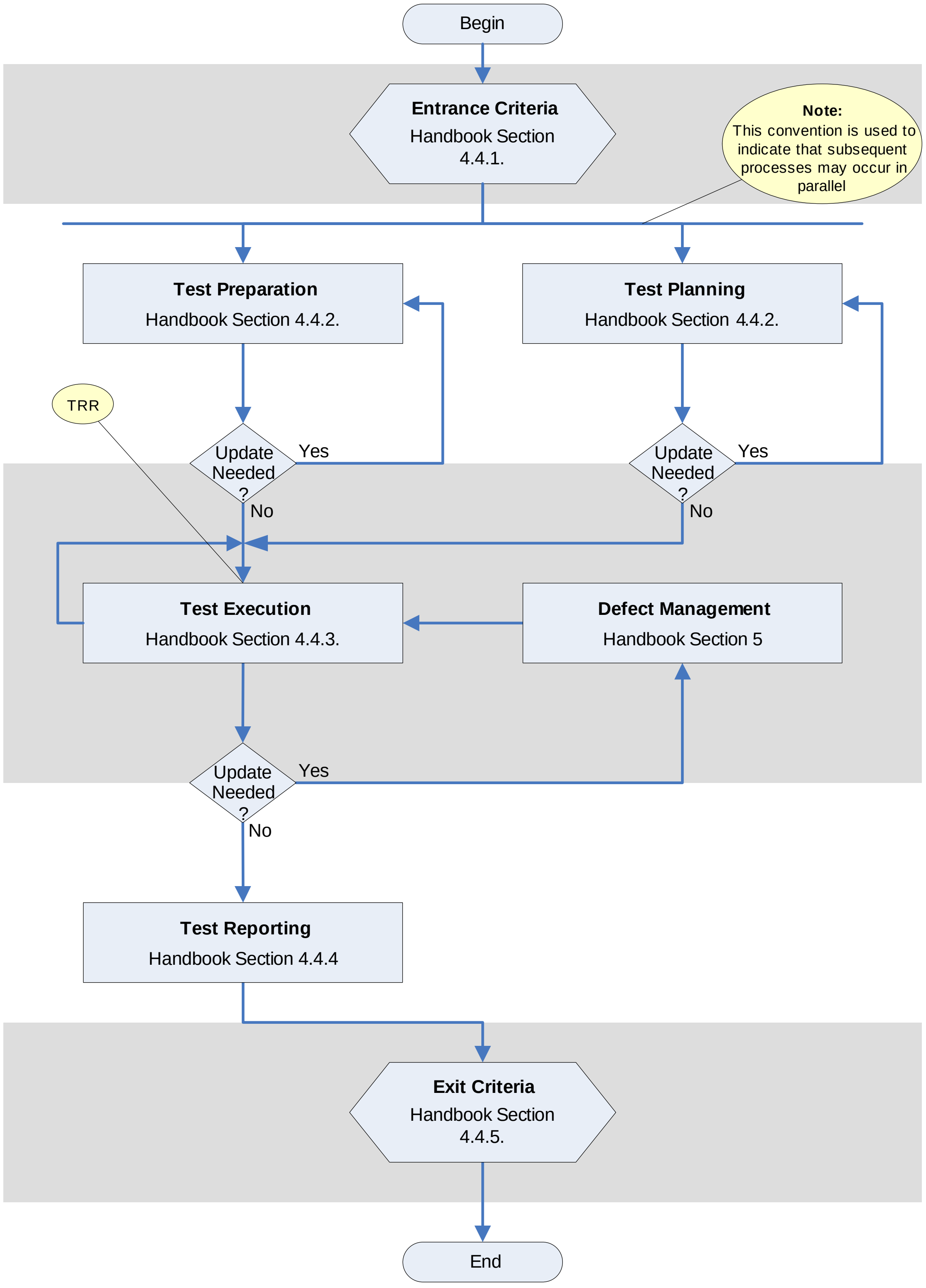
Role	Responsibility	Artifact(s)
Test Manager	Ensures that all Entrance Criteria are satisfied prior to beginning Test Preparation. Minimum Entrance Criteria include having: ▪ Baseline MTP ▪ Baseline Detailed Design Document ▪ Approved Requirements Traceability Matrix ▪ Approved System Test Plan ▪ System Test Environment ▪ Integration Testing completed ▪ System created by the developers that performed the integration ready to be tested ▪ Application login and password assigned ▪ Interface Control Document ▪ Coordination of interface testing activities ▪ The Test Readiness Review completed ▪ Tests assigned to testers ▪ All Test Suites, Test Cases, Test Scripts, and Test procedures created ▪ Completed a review of change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase ▪ Defect tracking tool is available and operational ▪ Works with the Federal Student Aid Project Manager, Contractor Project Manager, and Federal Student Aid Test Manager to evaluate the need to perform parallel System Testing and UAT ▪ Develops an appropriate mitigation strategy to ensure successful testing ▪ Follows up on outcomes of the TRR	▪ Approved Requirements Traceability Matrix ▪ System Test Plan created and approved ▪ System Test Environment exists ▪ Defect Reports ▪ Completed TRR ▪ Test Summary Report
	Oversees testing activities	▪ Email ▪ Meeting Minutes ▪ Status Reports
	Review all testing artifacts	▪ Reviewed System Test Cases ▪ Reviewed System Test Results

Role	Responsibility	Artifact(s)
	Ensure that all Exit Criteria are satisfied prior to submitting System Testing Artifacts to the Federal Student Aid Project Manager. Minimum Exit Criteria include ensuring that ▪ All planned System Test Cases pass ▪ All urgent and high priority defects identified during System Testing are resolved ▪ All artifacts, reports, and metrics are updated by the Test Team to reflect the current state of the tested system ▪ The Project Manager has reviewed and approved all artifacts ▪ Federal Student Aid Deliverable Review Process completed	▪ Completed System Test Cases ▪ Defect Log ▪ Updated artifacts, reports, and metrics ▪ Test Manager sign-off after reviewing and approving artifacts ▪ Federal Student Aid Test Manager sign-off after reviewing and approving artifacts
Federal Student Aid Test Manager	If a contract is involved, work with the Federal Student Aid Test Manager, and Contractor Project Manager, and Contractor Test Manager, to complete the following: ▪ Evaluate the need to perform parallel System Testing and UAT (performing testing in both phases concurrently) ▪ Develop an appropriate mitigation strategy to ensure that testing is successful ▪ Review test artifacts, provide feedback ▪ Communicate recommendation of acceptance or rejection of test artifacts	▪ Sign-off on test artifacts ▪ Comments matrix for reviewed test artifacts
Development Team	When testing with other applications is required, the development team must provide the Federal Student Aid test team the Interface Control Documents at least one week prior to the test effort unless the contract states otherwise. ▪ Perform defect analysis	▪ Interface Control Document ▪ Update to defect logs and impact analysis documentation

Role	Responsibility	Artifact(s)
System Test Team	Prepares System Testing by ▪ Updating the System Test Plan (as needed) ▪ Determining and obtaining the System Test Environment ▪ Determining and obtaining the System Test Framework required to execute System Test Cases ▪ Creating and updating System Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform each of the types of tests they will conduct during System Testing ▪ Obtaining logins and passwords to interface(s) ▪ Generating System Test data that: o Is reusable when appropriate o Is generated utilizing an automated tool whenever possible o Is generated from data that has been processed on the system and that can be re-used for testing	▪ Updated System Test Plan (if needed) ▪ System Test Framework ▪ System Test Suite (including test scenarios, test cases, test scripts, and test procedures) ▪ System Test Data
	Executes System Testing by ▪ Executing System Test Cases using the System Test Framework ▪ Evaluating System Test Results ▪ Performing Defect Management to verify, log and retest defects	▪ Executed System Test Suites ▪ Evaluated System Test results ▪ Complete documentation of defects in the defect management tool ▪ Updated code that resolves defects ▪ Documentation to show that defects were tracked and managed to the resolution
	Reports System Testing by ▪ Creating the System Testing Defect Report ▪ Submitting all artifacts to the Project Manager for review and approval	▪ Updated System Test Suites ▪ Updated System Test Plan ▪ System Test Summary Report ▪ System Test Metrics ▪ System Test Defect Report

The contractor responsible for System Testing will ensure that the test team tests the application to verify the functionality, performance, and reliability of the application according to approved design specifications.

Figure 4-4, Process Overview of System Testing



4.4.1 Entrance Criteria

The following conditions are the minimum Entrance Criteria that must be met before System Testing can begin.

- Baseline MTP
- Baseline Detailed Design Document
- Approved Requirements Traceability Matrix
- If applicable, Integration Testing is complete and all urgent and high errors found have been addressed and resolved
- The code deployed in environment by the developers that performed the integration is ready to be tested
- The application login and password have been assigned
- System Test Environment has been created
- System Test Plan has been created
- Test Readiness Review has been completed
- Tests have been assigned to Testers
- All Test Suites, Test Cases, Test Scripts, and Test Procedures have been created
- All change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase, have been reviewed
- Other criteria may be required based on the need of the project

These conditions must include consideration for each of the test types to be performed during System Testing.

4.4.2 Test Preparation

The System Testing Team is responsible for performing the following tasks before and, as needed, during System Testing.

- Updating the System Test Plan (as needed)
- Determining and obtaining the System Test Environment
- Determining and obtaining the System Test Framework required to execute System Test Suites
- Determining the guidelines between each application for Intersystem testing. The Test Manager or Project Manager of the application under test is responsible for working with the Test Manager and Project Manager of the other application to coordinate this testing phase
- Creating and updating System Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform each of the types of tests

- Obtaining logins and passwords to interface(s)
- Generating System Test data that:
  - Is reusable when appropriate
  - Is generated utilizing an automated tool whenever possible
  - Is generated from data that has been processed on the system and that can be re-used for testing

4.4.3 Test Execution

System Test execution tasks performed by the System Test Team include:

- Executing System Test Suites
- Evaluating System Test Results
- Performing Defect Management to verify, log and retest defects

Note: *Testing techniques useful during System Testing include, but are not limited to, Black Box Testing, Walkthrough, Regression Testing, Communication Testing, Security Testing, and Peer Review of Test Suites. The* ***Glossary in Appendix B*** *provides descriptions of each technique.*

If contractor testers are to perform Intersystem Testing during System Testing, files are created for transmission to business partners, and files are desk checked to ensure conformance to the recipient's formatting requirements. No data exchange will occur between the contractor and any party external to Federal Student Aid without the written authorization of the Federal Student Aid Project Manager.

If testing involves new applications or current applications containing new data interchange interfaces with stakeholders outside Federal Student Aid, exchange of data outside Federal Student Aid must not occur until Federal Student Aid has reviewed the application during UAT. The exception to this policy is that a limited exchange of data with a single partner may take place under Federal Student Aid supervision following receipt of written approval from the Federal Student Aid Project Manager. Federal Student Aid will exchange test files with partners during UAT to complete Intersystem Testing.

4.4.4 Test Reporting

System Test reporting includes, but is not limited to, creating the following artifacts:

- Updated System Test Suites
- Updated System Test Plans
- System Test Metrics
- Test Summary Report

- System Test Defect Report

4.4.5 Exit Criteria

Exit Criteria define quality standards that qualify the code for promotion to the next level or development stage. The following conditions are the minimum Exit Criteria for System Testing cases. (Federal Student Aid and/or the testing contractor may define additional Exit Criteria for a project – Exceptions must be approved by the Federal Student Aid Project Manager.)

- All System Test Cases have been executed.
- All urgent and high priority defects identified during System Testing are resolved. The Federal Student Aid Project Manager has discretion to add medium severities to this list.
- All the artifacts (including test cases and automated test scripts), reports, and metrics are updated by the testing team to reflect the current state of the interface(s).
- Test Manager has reviewed and approved all artifacts.
- Federal Student Aid Deliverable Review Process completed.
- Test coverage reported to Federal Student Aid.
- When a contractor is used, the Federal Student Aid Test Manager reviews the contractor's test results and accepts them, or requires additional testing.
- Other criteria may be required based on the needs of the project.

4.4.6 Performance Testing

The overall goal for any performance test project is to ensure that the application under testing operates as efficiently as possible. Performance testing is about balancing the needs of the customer with the resources available to the application owner.

The Federal Student Aid Project Manager must coordinate Performance Testing activities with the Enterprise Performance Testing Team in the early stages of a project. The Enterprise Performance Testing Team maintains a document that details the testing process.

Typically, the Performance Test Procedures are comprised of four phases. The phases are planning, development, execution, and documentation.

1. **Planning**
   - Discovery – the performance test team reviews requirements (business purposes, processes, and uses) and constraints of the system being tested, identifies stakeholders and their needs, determines the schedule, and determines the general usage patterns.
   - Requirements – the performance test team documents the efficiency of the application based on the best balance of the application owner's needs and the customers' needs. This allows the performance test team to set clearly defined goals for the test.
   - Scheduling – the performance test manager determines the schedule, specific tasks, resources needed and the length of time for each task required for a successful performance test. In addition to the performance test team, availability of the development and support staff must be considered.

2. Development

- Scripts – development of scripts depends on the business processes, which are then translated into steps for a virtual user to emulate a customer using the application during a performance test. The scripts selected must be those scenarios executed most frequently by the majority of users. All scripts are verified by peer reviews and mock load tests (10 users executing the script 10 times).
- Scenarios – development of scenarios is based on information from project documentation and subject matter experts. If the tool used for performance testing is Load Runner, user groups and load patterns are created.

3. Execution

- Testing – once the environment setup is complete, performance testing begins. An iterative approach is used during the test cycle. The scenarios and results are logged, development and support staff are notified of issues, changes may be made to the code or architecture, stakeholders are notified when test efforts are complete, and the daily *Test Summary Execution Report* (log) is published.

Daily meetings are held to: describe the results of the daily report, anomalies found, review of all issues, assignment of issues, assessment of impact and risks to the performance test schedule, which may impact the overall delivery schedule, identify and confirm the primary focus of the performance test, and agree on the strategy for the next performance test run.

- Tuning and Analysis – this process requires thorough analysis of data, teamwork (performance test team, project management, development team, and support staff) and constant communication. Views will be created for analyzing data, which may be web server health, application server health, application server session health, etc. The data should be created in a manner in which a novice will be able to review it and notice that something is amiss. All unexplained anomalies in the data must be identified, classified as problematic or symptomatic, and addressed. This is an iterative process. During this process, key metrics are assessed for performance, root cause of unexpected or undesirable behaviors of the system are analyzed, and all that has transpired during this process is documented.

Corrective action recommendations are presented to the stakeholders; Federal Student Aid project management approves the changes to be implemented.

- Reporting – the report provides information for peer review that can assist in validating the tasks performed as well as information about the problems encountered during the test and how they were resolved. Lastly, it provides guidance for improving the accuracy of future performance testing for the application under test. The report includes both failed and successful test runs. Keep in mind that additional information may be required based on the type of system undergoing performance testing.

Documenting results of the performance test is required. At a minimum, the report must contain the objective of the report that states why the performance test was needed, the 5 W's: (1) <u>who</u>: the members of the performance and development teams involved with the test; (2) <u>when</u>: what time each task was executed; (3) <u>where</u>: what was the state of the environment and the test suite where the test was run; (4) <u>why</u>: why was the task performed, what changes brought about the tasks and how the test was verified; and (5) <u>what</u>: what did the task accomplish, what were the resulting discoveries, changes and follow-up tasks and the conclusion stating whether or not the objectives of the test were met.

4. Documentation

- Readiness – this section of the final test summary report describes the expected behavior of the application under test at peak loads and identifies the point at which the application starts experiencing performance problems that may be due to availability of the system or transaction failures. This section gives management the information needed to deploy the application, which includes worst case scenarios, so they are prepared to address those issues if they occur. Bottom line, this section gives the go/no-go recommendation from the performance team based on the goals and the results of the test.
- Capacity – this section of the final test summary report describes the capacity planning done during the test and the run data on which it was based. Historical information on usage, projected usage, recorded data and monitoring information from peak runs from each run is included. Capacity based on run data and projected usage patterns and historical data are also included in this section.

The application development and support teams will review and accept this section.

This section is not mandatory and will be dictated by the type of project being tested.

- Configuration – this section of the final test summary report describes the steps to replicate the tuning done in the performance environment. Specific findings or anomalies are detailed and the corrective actions for each are included. All configuration details are also included. These corrective actions are not completed by the performance team but by the support team of the performance environment or the application development team.
- Findings – this section is the executive summary of the final test summary report, written with management in mind, and provides full disclosure of the facts. It describes the testing achieved, issues, and how the issues were resolved. The section will provide a chronology of the test and significant results, the worst and median of all the 90% response times, provide details on poor performing transactions, significant changes to the application or environment due to the findings, projected capacity, and list all the teams involved in the test effort.

The Federal Student Aid Performance Test Manager and the Performance Test Team approve the document prior to release to the Federal Student Aid Project Manager.

The primary testing requirement is to have an established, test ready environment that mirrors the environment of the system under test. Mirroring production provides the most accurate testing results, but due to complexity, budget restrictions, and testing goals, this may not always be possible. Test plans must include the environment expectations, and the environment must be validated before testing begins.

Planning for performance testing should start during the development of the system. However, since the system being developed is in flux, the planning will not be complete until after the initial test of the system is complete. A change in requirements may lead to a change in the business processes to be performance tested.

The performance test team must coordinate testing with the project manager and the development team. Performance testing should not begin until all features of the system have been functional tested, and there are no open issues or defects that impact the functionality of the system. Ideally, the component being tested should be finalized and have passed functional and user acceptance testing.

Parallel testing is allowed but must be agreed to by Federal Student Aid project management, and the following milestones must be completed: Performance Test Plan completed and approved, System Test of the system completed (if there is more than one cycle of System Testing, then the first cycle of system testing must be completed), and scripting of Performance Test Scenarios are completed.

Performance testing requires the active and responsible participation of members of the development and application support staff. In the final days of the performance test, developers are expected to be available full time to quickly address issues found during the performance test.

The application owner, development lead, and other stakeholders must approve the performance test plan.

- Typically, the performance test team will hold a meeting to discuss the plan and may require designated stakeholders to be physically present at the meeting.

The performance test schedule must be completed and approved by stakeholders prior to the start of the performance test.

Federal Student Aid project management must give the performance test team the green light to proceed with the test.

Performance Testing is an exception to the process described as System Testing. The high-level differences are described below.

4.4.6.1 Entrance Criteria

The following conditions must exist before Performance Testing can occur:

- Baseline software requirement specification, functional requirement document, detailed system design documents, and Master Test plan and/or Intersystem Test plan have been created
- The application to be tested has been tested for system functionality
- A performance testing tool has been installed
- A Test Environment has been created for Performance Testing
- All the performance test scripts complete successfully
- The Performance Test Plan has been created

4.4.6.2 Test Preparation

The application team provides test data for the performance test. The type of data needed may include:

- Test data for boundary value analysis and equivalence partitioning
- Test data which considers volumes required and anticipated usage patterns, as well as data dependency

4.4.6.3 Performance Test Execution

The Performance Test Team is responsible for performing the following Performance Test related tasks:

- Reviewing Performance Test readiness
- Creating Performance Test Suites, scenarios, Test cases, scripts and procedures
- Determining the monitoring metrics to be measured, maximum allowable values, anticipated values, and minimum acceptable values. *Note: Input will be required from the project subject matter experts.*
- Executing the Performance Test Scripts
- Validating Performance Test results
- Tracking and managing defects
- Creating Performance Test Reports consisting of Performance Test metrics

4.4.6.4 Performance Test Reporting

Performance Testing must provide the following outputs:

- Performance Test Summary Report
- Performance Test Defect Report
- List of configuration changes needed on the application and on the environment

4.4.6.5 Exit Criteria

The Exit Criteria for Performance Test Cases are:

- Application is stable under a specified load
- Necessary configuration changes have been applied

4.5 Phase 4: User Acceptance Testing (UAT)

Federal Student Aid Application Owners perform UAT. UAT validates requirements and determines whether the application operates in the Federal Student Aid business environment. Examples of these applications include web-based applications, on-line mainframe (Customer Information Control System [CICS]) applications, COTS applications such as SIEBEL, reporting tools, and others where users may touch and feel the application. UAT focuses on requirements and application functionality.

The test contractor may be responsible for developing the UAT Test Plan, Test Suites, and related UAT activities and documentation. In all cases, the test contractor is responsible for providing data to be used for UAT.

If the test contractor is not contractually responsible for these tasks and for assisting Federal Student Aid in performing UAT, the Federal Student Aid Test Manager is responsible for performing these tasks. The testing types that occur during UAT are:

- Usability Testing (Required)
- If applicable, Intersystem Testing (Required)
- Regression Testing (Required)
- Functional Testing (Required)
- Specialized Testing Types if applicable:
    - 508 Compliance Testing
    - Security Testing

The UAT Team, including the Test Manager and UAT Testers, plan UAT activities. The UAT Test Plan includes the assigned tasks, schedule, and resources. Planning must include review of Integration Testing Defect Reports and System Testing Defect Reports to identify problems and areas of concern that occurred in previous testing phases and that may need additional scrutiny during UAT. When UAT includes Intersystem Testing, the Federal Student Aid Test Manager is responsible for coordinating testing activities with the major players of subsystems that may be affected by the testing activities.

For applications that include functionality performed through batch jobs, the contractor is responsible for running each job and providing Federal Student Aid with the data used to create the output and with the hard copy reports and/or files produced. Batch functionality typically includes processing data received from other systems, creating data for transmissions to other systems, and updating data without any manual modification of data. Federal Student Aid will be responsible for validating the output from those jobs.

Example – User Acceptance Testing

UAT Scenario

The Federal Student Aid Information Center's Interactive Voice Response Unit (IVRU) allows callers to speak the first two letters of their last name to access loan status information.

Test

The Application Owner tester calls into the Federal Student Aid Information Center and receives a prompt to speak the first two letters of the last name being used for the test. The system does not recognize the letters spoken by the tester on the first two attempts. The system then recognizes the letters spoken on the third attempt. The Application Owner would then determine if the IVRU system met the acceptable quality level, business expectations, and/or service level agreements for the project.

When an application provides functionality such as data entry, data update, and data retrieval on-line or through a reporting tool to Application Owners within Federal Student Aid, Federal Student Aid is responsible for performing hands-on UAT.

Table 4-4, UAT Testing Roles, Responsibilities, and Artifacts, shows the Project Manager, Test Manager, and UAT Test Team roles and responsibilities as required by Federal Student Aid, and the artifacts that must be created for successful UAT.

Test suites that have been approved by Federal Student Aid may not be changed during test execution. When analysis of defects leads to requirement clarification or modification, then new Test Suites must be created and approved by Federal Student Aid.

Table 4-4, UAT Testing Roles, Responsibilities, and Artifacts

Role	Responsibility	Artifact(s)
Federal Student Aid Project Manager	Works with the Federal Student Aid Test Manager, Contractor Project Manager, and Contractor Test Manager if a contract is involved, to evaluate the need to perform parallel System Testing and UAT (performing testing in both phases simultaneously) for part of each testing phase and to develop an appropriate mitigation strategy to ensure that testing is successful. The Federal Student Aid Project Manager will approve or reject a recommendation to perform testing using this technique, which should only be used on an exceptional basis. Provides TRR approval.	▪ Approved parallel testing plan and related mitigation strategy ▪ Approved test artifacts

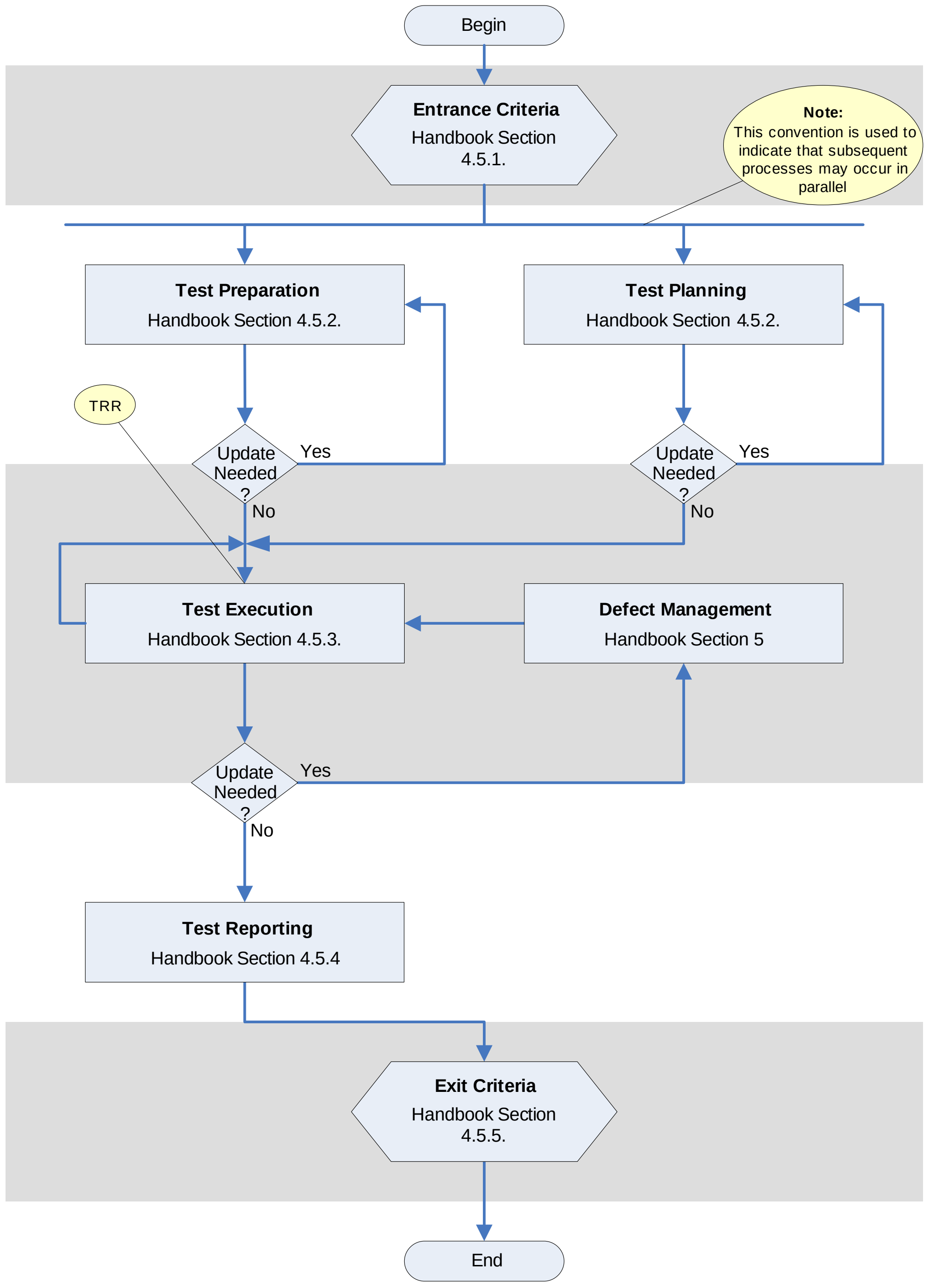
Role	Responsibility	Artifact(s)
Contract Project Manager	Works with the Federal Student Aid Project Manager, Federal Student Aid Test Manager, and Contractor Test Manager to evaluate the need to perform parallel System Testing and UAT (performing testing in both phases simultaneously) for part of each testing phase and to develop an appropriate mitigation strategy to ensure that testing is successful.	▪ Approved test artifacts
	Provides information for testing and testing related activities. Provides TRR approval.	▪ Updated Project Development Plan ▪ Updated Work Breakdown Structure
Federal Student Aid Test Manager	Works with the Federal Student Aid Project Manager, Contractor Project Manager, and Contractor Test Manager if a contract is involved, to evaluate the need to perform parallel System Testing and UAT (performing testing in both phases simultaneously) for part of each testing phase and to develop an appropriate mitigation strategy to ensure that testing is successful.	▪ Approved parallel testing plan and related mitigation strategy
	Reviews Deliverables from test effort. Provides TRR recommendation.	▪ Comments Matrix for testing artifacts ▪ Sign-off after reviewing and approving artifacts
Test Manager	Works with the Federal Student Aid Project Manager, Contractor Project Manager, and Federal Student Aid Test Manager to evaluate the need to perform parallel System Testing and UAT (performing testing in both phases simultaneously). If such a technique is to be recommended, the Test manager will develop the recommendation, and develop an appropriate mitigation strategy to ensure that testing is successful even if testing problems occur employing this technique.	▪ Approved testing artifacts ▪ Comments matrix for testing artifacts

Role	Responsibility	Artifact(s)
	Ensures that all Entrance Criteria are satisfied prior to beginning Test Preparation. Minimum Entrance Criteria include having: ▪ Baseline MTP ▪ Baseline Detailed Design Document ▪ Approved Requirements Traceability Matrix ▪ Approved UAT Test Plan ▪ UAT Test Environment ▪ Application software installed ▪ Application login and password assigned ▪ UAT Plan ▪ TRR documentation and process ▪ System Testing completed with all urgent and high errors having been resolved ▪ Tests assigned to testers ▪ All Test Suites, Test Cases, Test Scripts, and Test Procedures created ▪ Completed a review of change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase ▪ Generated UAT Test data that: o Is reusable when appropriate o Is generated utilizing an automated tool whenever possible o Is generated from data that has been processed on the UAT and that can be re-used for testing o Reviews test status from testers which may include sign-off sheets o Follows up on outcomes of the TRR	▪ UAT Test Plan created and approved ▪ UAT Test Environment exists ▪ Integrated software components ▪ User ID and password ▪ UAT Support Test Plan ▪ Completed TRR ▪ System Test Defect Report ▪ Tests have been assigned to Testers ▪ All Test Suites, Test Cases, Test Scripts and Test Procedures have been created ▪ All change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase have been reviewed ▪ UAT Test Data

Role	Responsibility	Artifact(s)
	Oversees testing activities.	▪ Email ▪ Meeting Minutes ▪ Status Reports
	Reviews all testing artifacts.	▪ Reviewed UAT Test Cases ▪ Reviewed UAT Test Results
	Ensures that all Exit Criteria are satisfied prior to submitting UAT Artifacts to the Federal Student Aid Project Manager. Minimum Exit Criteria include ensuring that: ▪ All planned UAT Test Suites pass ▪ All urgent and high priority defects identified during UAT are resolved ▪ All artifacts, reports, and metrics are updated by the Test Team to reflect the current state of the UAT ▪ The Project Manager has reviewed and approved all artifacts ▪ The Federal Student Aid Deliverable Review Process is completed ▪ When a contractor is used, the Federal Student Aid Test Manager reviews the contractor's test results and accepts them, or requires additional testing	▪ Completed UAT Test Cases ▪ Defect Log ▪ Updated artifacts, reports, and metrics ▪ Test Manager sign-off after reviewing and approving artifacts ▪ Federal Student Aid Test Manager sign-off after reviewing and approving artifacts ▪ Test Coverage reported
Development Team	When testing with other applications is required, the development team must provide the Federal Student Aid test team the Interface Control Documents at least one week prior to the test effort unless the contract states otherwise.	▪ Interface Control Documents ▪ Data Management Information
UAT Test Team	UAT Testing preparation ▪ Updating the UAT Test Plan (as needed) ▪ Determining and obtaining the UAT Test Environment required to execute UAT Test Suites ▪ Creating and updating UAT Test Suites, including determining test scenarios, test scripts, and test procedures to perform each of the types of tests they will conduct during UAT ▪ Obtaining logins and passwords	▪ Updated UAT Test Plan ▪ UAT Test Framework ▪ UAT Test Suite (including test scenarios, test scripts, and test procedures)

Role	Responsibility	Artifact(s)
	UAT execution by: ▪ Executing UAT Test Cases using the UAT Test Framework ▪ Evaluating UAT Test Results ▪ Performing Defect Management to verify, log and retest defects	▪ Executed UAT Test Cases ▪ Evaluated UAT Test results ▪ Defects entered into the defect management tool ▪ Updated code that resolves defects ▪ Defects tracked and managed to the resolution
	UAT reporting by: ▪ Updating UAT Test Cases ▪ Updating UAT Plans ▪ Creating UAT Reports ▪ Creating UAT Metrics ▪ Creating the UAT Defect Report	▪ Updated UAT Test Suites ▪ Updated UAT Test Plan ▪ UAT Test Summary Report ▪ UAT Test Metrics ▪ UAT Test Defect Report

Figure 4-5, Process Overview of User Acceptance Testing



4.5.1 Entrance Criteria

The following conditions are the minimums that must be met before UAT can begin. Federal Student Aid may define additional Entrance Criteria for a project. A contractor may recommend additional criteria for Federal Student Aid approval. The Federal Student Aid Project Manager must approve exceptions.

- Baseline MTP
- Baseline Detailed Design Document
- Approved Requirements Traceability Matrix
- Approved UAT Plan
- UAT environment
- Application software installed
- Application login and password have been assigned
- UAT Support Plan has been created
- TRR Completed
- System Testing is complete and all urgent and high errors found have been addressed and resolved
- Tests have been assigned to Testers, and all Test Suites, Test Cases, Test Scripts and Test Procedures have been created

These conditions must include consideration for each of the test types to be performed during UAT.

4.5.2 Test Preparation

Before UAT can begin, the UAT Team must make the following test preparations

- Updating the UAT Test Plan (as needed)
- Determining and obtaining the UAT Test Environment required to execute UAT Test Suites
- Creating and updating UAT Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform each of the types of tests. The contractor develops UAT Test Suites unless stated otherwise in the contract.
- Obtaining logins and passwords
- Generating UAT Test data that:
  - Is reusable when appropriate
  - Is generated utilizing an automated tool whenever possible

  - Is generated from data that has been processed on the UAT and that can be re-used for testing

- The contractor develops UAT test data and is responsible for providing the data to Federal Student Aid in accordance with the project schedule.

4.5.3 Test Execution

Once the test preparations are completed, the UAT Test Team performs the actual User Acceptance, as follows:

- Executing UAT Test Suites using the UAT Test Framework
- Evaluating UAT Test Results
- Performing Defect Management to verify, log, and retest defects

Note: *Testing techniques that are useful during UAT include, but are not limited to, black box testing, regression testing, 508 testing, communication testing, and security testing. The* ***Glossary in Appendix B*** *provides descriptions of each technique.*

If contractor testers are to perform Intersystem Testing during UAT, files are created for transmission to business partners, and files are desk checked to ensure conformance to the recipient's formatting requirements. No data exchange will occur between the contractor and any party external to Federal Student Aid without the written authorization of the Federal Student Aid Project Manager.

4.5.4 Test Reporting

The UAT Testing Team prepares the following reports and submits them to the Test Manager or Project Manager for review and approval. Additional reports may be required for a specific project.

- UAT Test Suites
- UAT Plans
- UAT Reports
- UAT Metrics
- UAT Defect Report
- UAT Sign-Off Sheet (template in **Appendix E – Templates)** when required by the Test Manager

4.5.5 Exit Criteria

UAT will not be considered complete until all of the Exit Criteria are completed and approved. Federal Student Aid may define additional Exit Criteria for a project. A contractor may recommend additional criteria for Federal Student Aid approval. The Federal Student Aid Test Manager must approve exceptions. The following are the minimum Exit Criteria:

- All urgent and high priority defects (and all other defects as defined by the Federal Student Aid Project Manager) identified during UAT are resolved
- All the artifacts (including test cases and automated test scripts), reports, and metrics are updated by the testing team to reflect the current state of the UAT
- The Test Manager has reviewed and approved all artifacts
- The Federal Student Aid Deliverable Review Process is complete
- Test Summary Report delivered to Federal Student Aid
- UAT Sign-Off Sheets complete

4.6 Phase 5: Post Implementation Verification (PIV)

The last testing phase is Post Implementation Verification (PIV), which occurs after the application is in production. Three specific types of testing may occur during PIV:

1. Post Implementation Verification
2. First Live Batch Testing
3. Post Implementation Support Testing (discussed separately in subsection 4.7)

The objective of **Post Implementation Verification** is to validate the results of critical functions as determined by the application owner. The validation occurs in the production environment when the function is naturally scheduled or a special execution of the function may be required.

An example of Post Verification testing is to verify that the monthly SAIG report contains accurate data. The install may occur on the 10th of the month, but the monthly report is scheduled to run on the 28th of the month. The Post Implementation Verification testing will occur on the 28th of that month.

The objective of **First Live Batch Testing** is to validate the execution of critical functions of the application when the application is installed into production. If the test is found to be unacceptable, the application owner has the option of requiring the software be rolled back. At that point, testing must take place in production to ensure that the correct version was installed. Testing must also occur in the test environment to help in the analysis of the defect that was found during First Live Batch.

An example of First Live Batch testing is to verify that records for Pell Grant awards appear on the web. Schools will be prevented from sending data until the first live batch test is proved successful. One school will be allowed to send data to Federal Student Aid that will in turn trigger the application to update the Pell Grant awards data. If the web does not display the expected results, the application owner may require the software to be rolled back to its previous version.

Table 4-5, First Live Batch Testing Roles, Responsibilities, and Artifacts, summarizes Federal Student Aid's minimum requirements for the roles, responsibilities, and artifacts related to First Live Batch Testing.

Test Suites that have been approved by Federal Student Aid may not be changed during test execution. When analysis of defects leads to requirement clarification or modification, then new Test Suites must be created and approved by Federal Student Aid.

Table 4-5, First Live Batch Testing Roles, Responsibilities, and Artifacts

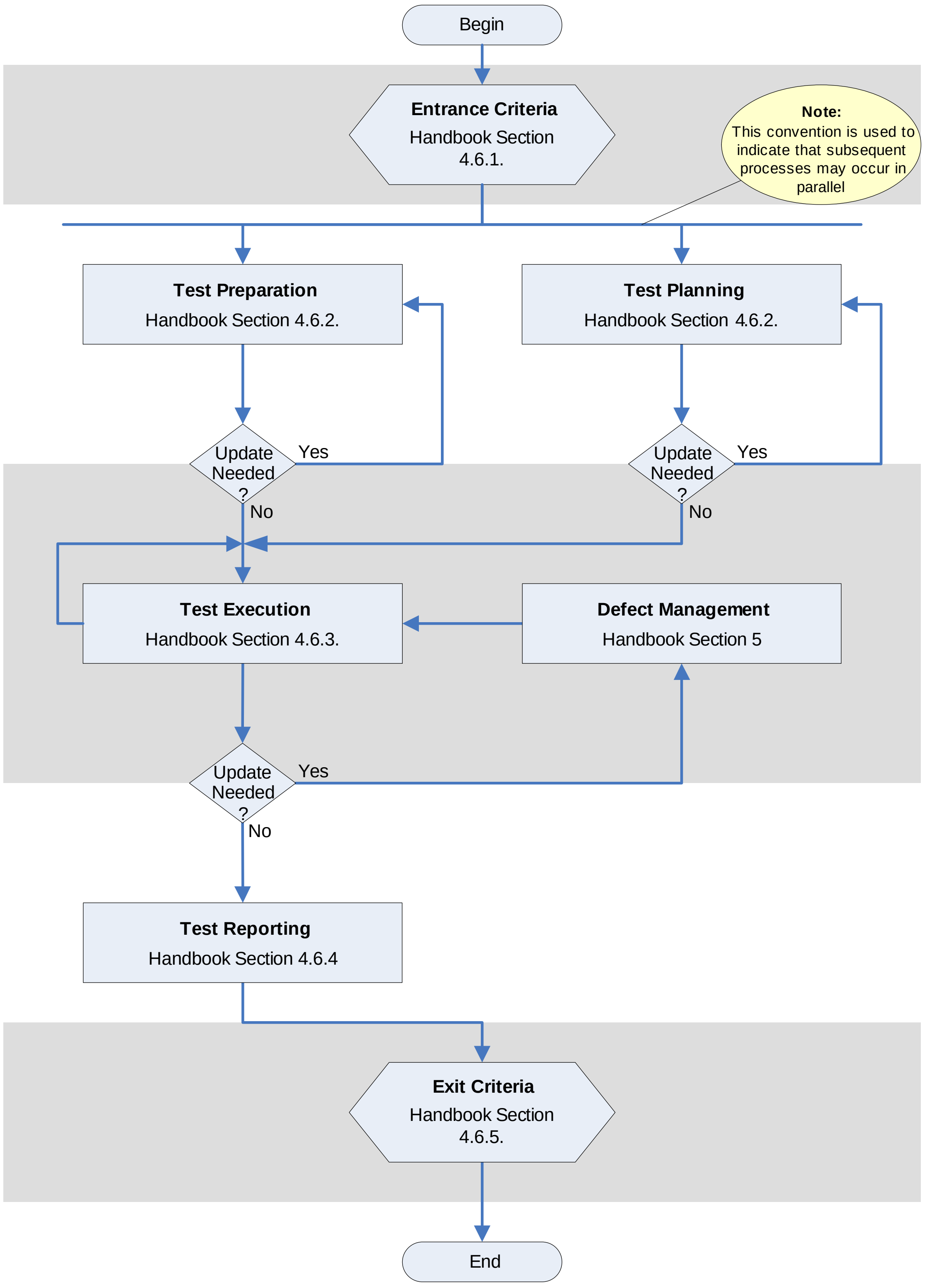
Role	Responsibility	Artifact(s)
Federal Student Aid Project Manager	Provides information for testing and testing related activities.	▪ Updated Project Development Plan ▪ Updated Work Breakdown Structure
	Shares and vets the PIV test plan with the Operations and Maintenance contractor.	▪ None
Test Manager	Ensures that all Entrance Criteria are satisfied prior to beginning Test Preparation. Minimum Entrance Criteria include having ▪ The application deployed to a production environment ▪ Conducted a Post-Implementation Review (PIR) to review and assess if the production application has been documented ▪ Post Implementation Verification (test) Plan created ▪ Baseline Design Document ▪ Baseline MTP ▪ Baseline Detailed Design Document ▪ Approved Requirements Traceability Matrix ▪ Application login and password assigned ▪ Tests assigned to testers ▪ Test Suites, Test Cases, Test Scripts, and Test Procedures created ▪ Completed a review of change requests received since the last baseline of the requirements and design documents, and artifacts from the previous test phase	▪ Completed TRR ▪ Baseline MTP ▪ Baseline Detailed Design Document ▪ Approved Requirements Traceability Matrix ▪ First Live Batch Testing Test Plan created and approved ▪ First Live Batch Testing Test Environment exists ▪ Integrated software components
	Oversees testing activities.	▪ Email ▪ Meeting Minutes ▪ Status Reports

Role	Responsibility	Artifact(s)
	Reviews all testing artifacts.	▪ Reviewed First Live Batch Testing Test Cases ▪ Reviewed First Live Batch Testing Test Results
	Ensures that all Exit Criteria are satisfied prior to submitting First Live Batch Testing Artifacts to the Federal Student Aid Project Manager. Minimum Exit Criteria include having: ▪ The functions tested during First Live Batch Testing work correctly ▪ All First Live Batch Testing cases passed ▪ All urgent and high priority defects encountered in the First Live Batch Testing phase reported and resolved ▪ The Federal Student Aid Test Manager working in consultation with the contractors, Federal Student Aid Application Owners, and project teams to determine whether the First Live Batch Testing test results are satisfactory. ▪ All artifacts, reports, and metrics updated by the Test Team to reflect the current state of the tested system ▪ The Project Manager completing the review and approving all artifacts ▪ The Federal Student Aid Deliverable Review Process completed ▪ When a contractor is used, the Federal Student Aid Test Manager reviews the contractor's test results and accepts them, or requires additional testing	▪ Completed First Live Batch Testing Test Cases ▪ Defect Log ▪ Updated artifacts, reports, and metrics ▪ Test Manager sign-off after reviewing and approving artifacts ▪ Federal Student Aid Test Manager sign-off after reviewing and approving artifacts

Role	Responsibility	Artifact(s)
First Live Batch Testing/PIV Test Team	First Live Batch Testing preparation ▪ Creating First Live Batch Testing Test Suites and determining the First Live Batch Testing Functional Test scenarios consisting of the First Live Batch Testing Functional Test Suites, test Scripts, and test Procedures. ▪ Creating First Live Batch Testing test reports that include test metrics and test results ▪ Using production data to perform First Live Batch Testing Functional Tests. ▪ Updating the Post Implementation Verification Plan (as needed) ▪ Creating and updating First Live Batch Test Suites, including determining test scenarios, test cases, test scripts, and test procedures to perform each of the types of tests they will conduct during First Live Batch	▪ Updated First Live Batch Testing Test Plan (if needed) ▪ First Live Batch Testing Test Framework ▪ First Live Batch Testing Test Suite (including test scenarios, test cases, test scripts, and test procedures ▪ First Live Batch Testing Test Data
	First Live Batch Testing execution by: ▪ Executing First Live Batch Test Suites using the Production system ▪ Evaluating First Live Batch Test Results ▪ Performing Defect Management to verify, log, and retest defects	▪ Executed First Live Batch Testing Test Suites ▪ Evaluated First Live Batch Testing Test results ▪ Defects entered into the defect management tool ▪ Defects tracked and managed to the resolution
	First Live Batch Testing/PIV Testing reporting by ▪ Creating First Live Batch Testing Summary Reports ▪ Creating First Live Batch Testing metrics ▪ Creating First Live Batch Testing Defect Reports, if applicable ▪ Submitting all artifacts to the Project Manager for review and approval	▪ Updated First Live Batch Testing Test Suites ▪ Updated First Live Batch Testing Test Plan ▪ First Live Batch Testing Test Summary Report ▪ First Live Batch Testing Test Metrics ▪ First Live Batch Testing Test Defect Report

The contractor is responsible for First Live Batch Testing unless the contract indicates otherwise. The responsible party will ensure that the application's functionality, performance, and reliability meet Federal Student Aid's requirements.

Figure 4-6, Process Overview of Post Implementation Verification



4.6.1 Entrance Criteria

The following conditions must be met before First Live Batch Testing can occur:

- The application must be deployed to a production environment
- A PIR must be conducted and documented
- A PIV (test) Plan must be created
- Baseline Design Document must be created
- Baseline MTP must be created
- Baseline Detailed Design Document must be created
- Approved Requirements Traceability Matrix must be created
- Application login and password must be assigned
- Tests must be assigned to testers
- Test Suites, Test Scripts, and Test Procedures must be created
- A review of change requests received since the last baseline of the requirements and design documents must be created, and artifacts from the previous test phase must be developed

4.6.2 First Live Batch Testing Preparation

The Post Implementation Verification team is responsible for performing the following tasks before and during First Live Batch Testing:

- Creating First Live Batch Test Suites and determining the First Live Batch Testing Functional Test scenarios
- Creating First Live Batch Test reports that include test metrics and test results
- Updating the PIV Plan (as needed)
- Obtaining logins and passwords to interface(s)

4.6.3 Test Execution

The Test Team performs the following execution activities:

- Executing First Live Batch Test Suites in the Production environment
- Evaluating First Live Batch Test Results
- Performing Defect Management to verify, log and retest defects

4.6.4 Test Reporting

Post Implementation Verification must provide the following outputs:

- First Live Batch Test Summary Reports

- First Live Batch Test Metrics
- First Live Batch Test Defect Reports, if applicable
- Submitting all artifacts to the Project Manager for review and approval

4.6.5 Exit Criteria

Exit Criteria define the standards for work product quality that enables it to go to the next level or phase of testing. The following Exit Criteria are required for PIV cases:

- The critical functions tested during First Live Batch Testing work correctly
- All First Live Batch Testing cases passed
- All urgent and high priority defects or others as defined by the Federal Student Aid Project Manager encountered in the First Live Batch Testing phase are reported and resolved
- The Federal Student Aid Test Manager, working in consultation with the contractors, the Federal Student Aid Application Owner, and project teams to determine whether the First Live Batch Testing test results are satisfactory
- All artifacts, reports, and metrics updated by the Test Team reflect the current state of the tested system
- The Project Manager completing the review and approving all artifacts
- The Federal Student Aid Deliverable Review Process completed
- When a contractor is used, the Federal Student Aid Test Manager reviewing the contractor's test results and accepting them, or requiring additional testing

4.7 Post Implementation Support Period

The Post Implementation Support Period (PISP) is the time period after implementation of the application into the production environment when the development contractor is responsible for defect resolution. In some commercial industries and federal agencies, the PISP is often referred to as the warranty period.

The application development contract must specify the term of the PISP, and the testing contract must require that testing resources are available during the support period. This will ensure that the development or maintenance activities receive the same level of testing as is given prior to acceptance of the application by Federal Student Aid. The combined contractor and Federal Student Aid Change Control Board (CCB) will prioritize the defects identified during the Post Implementation Support Period. The defects will then be modified by the development team and tested by the contractor and Federal Student Aid testers. Metrics are captured during this test period.

4.8 Testing Emergency Software Releases

- Urgent and high errors found in production require the immediate attention of the contractor and the Federal Student Aid Application Owner. Testing support must be available when an urgent and high error is found in the application. The Test Team will need to review the requirements and design test suites to address the corrective application modification. Testing must include regression test scenarios to ensure that key functionality has not been compromised. Testing must be accurate to confirm that the modification meets the business need.

Section 5. Defect Management

5.1 Overview

Defect Management is the process of recognizing, investigating, taking action on, and disposing of defects. Defect Management follows the testing phase activities. Since correcting defects is both time consuming and expensive, defect prevention is desirable. Although prevention is mainly the responsibility of the development team, the testing team is also responsible to ensure that test scenarios, scripts and the test environment are correct to avoid wasting time on invalid defects.

Defects must be identified and resolved through a specific Defect Management Process, which includes:

- **Defect Prevention**: A risk based process involving the identification of urgent and high defect related risks, estimation of the impact of each of the risks, and using risk mitigation to minimize the impact of each risk.
- **Defect Discovery**: Includes identifying defects, documenting them, and receiving acknowledgement from the development team that the Testing Team has identified a valid defect.
- **Defect Tracking**: The Testing Team enters each defect into a Defect Management Tool as soon as the defect is discovered, and tracks each defect to closure.
- **Defect Correction or Resolution:** Following discovery, developers prioritize, schedule, resolve a defect, document the resolution, and provide notification of the resolution so the tester can validate that the issue has been resolved based on assigned severity and priority. .
- **Process Improvement**: Analysis of the process in which a defect, found in testing, originated to identify ways to improve the process to prevent future occurrences of similar defects.

During the project and at the end of the project, the Test Manager reports defects and related metrics to the contract Project Manager, the Federal Student Aid Project Manager, and the Federal Student Aid Enterprise Test Manager. To provide maximum benefit, the defect management reporting process must include metrics for successes and failures. Federal Student Aid will use these reports in process improvement efforts. The overall goal is to use lessons learned either to minimize defects in the future, or to identify defects early in the development process in order to reduce adverse impact and cost of correction.

A deliverable is baselined each time a predefined milestone is reached. Errors identified before a deliverable is baselined will not be considered a defect.

At the end of each test phase all defects must either be closed, deferred or in a state which is acceptable to the Federal Student Aid Test Manager/Test Lead.

5.2 Relevance of Defect Management to the Handbook Audience

The following key staff members need to fully understand and follow the requirements in the Defects Management section of this Handbook. Subsections following this list describe the responsibilities for each of the following positions:

- Federal Student Aid and/or Contractor Project Manager
- Federal Student Aid and/or Contractor Test Manager/Test Lead
- Federal Student Aid Enterprise Test Manager
- Federal Student Aid Application Owner
- Federal Student Aid and/or Contractor Application Development Team
- Federal Student Aid and/or Contractor Application Test Team

Table 5-1, Defect Management Roles & Responsibilities

Title	Responsibilities
Project Manager	▪ Analyzing defect metrics and reports and providing input to the defect prevention and resolution process. ▪ Receiving information throughout the defect lifecycle including defect evaluation information, defect status information, and defect resolution information. ▪ Being involved in assigning defect priorities.
Test Managers	▪ Analyzing management reports and metrics, and providing input to the defect prevention, defect discovery, and process improvement processes. ▪ Selecting a defect-tracking tool.
Federal Student Aid Enterprise Test Management	▪ Providing input to the defect prevention effort. ▪ Analyzing defect data for trends. ▪ Defect resolution and process improvement processes through various management reports. ▪ Validating compliance of the testing contractor with Federal Student Aid Defect Management Requirements.
Federal Student Aid Application Owners	▪ Reporting defects discovered during UAT. ▪ Tracking defects to resolution. ▪ Providing input into the defect prevention, defect discovery and process improvement processes using information obtained from Defect Management reports.
Application Development Teams	▪ Resolving defects reported during each phase of testing. ▪ Deliverable baselining. ▪ Defect analysis. ▪ Providing input into the defect prevention, defect discovery and process improvement processes using information obtained from Defect Management reports. ▪ Recording or updating defects in a defect tracking tool.

5.3 Defect Classification

A valid defect is one that is not a duplicate of a previously reported defect and one that can be reproduced. The following defect classifications will be used during the test effort:

- Type
- Severity
- Complexity
- Priority
- Status

5.3.1 Defect Types

A defect type indicates the source and/or location of the defect. The test team may assign multiple defect types to a single defect. Defect types include the following characteristics:

- Gap in requirements definition
- Design problem
- Data element problem (missing, incorrect edit, etc.)
- Calculation problem (missing, incorrect result, etc.)
- User Interface (incomplete, incorrect presentation, etc.)
- Intersystem Interface (file layout, etc.)
- Report (missing data, incorrect format, etc.)
- New Requirement (an enhancement or may have missed something in the development process that will need to be fixed and issued in the next release)
- Tester Error/Invalid Defect

5.3.2 Defect Severity

Defect severity indicates the level of business impact of the defect. The tester defines defect severity using one of four levels of severity based on the IEEE scheme.

Table 5-2, Defect Severity Levels

Severity	Description	Example
Urgent	Prevents the accomplishment of an operational or mission essential capability.	• Jeopardizes security (e.g., potential violation of Privacy Act) • Jeopardizes data quality (e.g., corrupt data entered into the system, data is corrupted by the system, or data is calculated incorrectly) • Software crashes (deprive user of system or cause loss of data) • Loss of data (e.g., incorrect file formats

Severity	Description	Example
		sent to other systems or decoded incorrectly) • Jeopardizes safety (normally not software related, but may be, e.g. if personal safety is involved)
High	Adversely affects the accomplishment of an operational or mission essential capability and no work around solution is known.	• Noncompliance with a required standard (e.g., 508 Compliance if required) • Software issues which are not critical but impact the use of the system (e.g., printing errors, inability to print a form) • Data input problems (which do not corrupt data), etc. • Missing help files, no other system documentation • Missing windows • Missing functionality that is not crucial • A search function that provides incorrect results
Medium	Adversely affects the accomplishment of an operational or mission essential capability, but a work around solution is known and productivity is negatively impacted.	• A required help file is missing, but there is a hardcopy document containing the same data available to the user. • A form or window does not print, but there is another way to print it (e.g., the print button does not work, but there is a pop-up window which has a print function and it works)
Low	Results in user inconvenience or annoyance but does not affect a required operational or mission essential capability.	• Typos in help files • Minor color issues such as window or button colors (which do not impact 508 compliance if required) • Typos on screen for internal Federal Student Aid use only (user facing errors should be rated high or medium) • Typos in documentation for internal Federal Student Aid use only (user facing errors should be rated high or medium) • Operator annoyances (e.g., tabbing between fields is not in the most desirable order or sub-window groupings are not logical)

5.3.3 Defect Complexity

Defect complexity provides an estimate of the difficulty/risk of the change of a component or environment. The Development Team is responsible for assigning one of the three levels of complexity: high, medium, and low.

Table 5-3, Defect Complexity Levels

Level	Description
High	Complex changes such as altering the logic of one or more components or the structure of the document will require significant effort (e.g., more than 40 man hours).
Medium	Simple changes requiring minimal effort (e.g., 40 man hours or less).
Low	No change to the logic of the component or the structure of the document.

5.3.4 Defect Priority

Defect priority indicates the order in which the development team will address defects. This rating provides an estimate of the relative standing of the issue in relation to all other identified defects. The Defect Priority categories are:

- **High**: All Urgent and most High severity defects
- **Medium**: High severity defects not assigned to a high priority
- **Low**: All Medium and Low severity defects

The Federal Student Aid Test Manager and Federal Student Aid Project Manager have the authority to override the priority assigned by the contractor. The Federal Student Aid Test Manager assigns defect priority during the UAT phase.

5.3.5 Defect Status

Defect status indicates the current disposition of a defect. There are eight defect states: open, evaluated, closed, deferred, assigned, analyzed, corrected and not scheduled for retesting, and corrected and scheduled for retesting.

Table 5-4, Defect Status Levels

#	Defect	Description
1.	**Open**	A submitted defect.
2.	**Evaluated**	A submitted defect that has been reviewed for validity.
3.	**Closed**	A defect that has been determined to be a duplicate, not valid, or passed due to a retest and the defect no longer needs to be managed.
4.	**Deferred**	A defect where a decision was made to postpone the resolution of the defect.
5.	**Assigned**	A defect where a developer has been assigned to analyze and resolve

#	Defect	Description
		the defect.
6.	**Analyzed**	A defect where a developer and a tester have reviewed the defect to determine the impact of resolving the defect on the project.
7.	**Corrected and not Scheduled for Retesting**	A defect that has been corrected by the developer and is ready for retesting.
8.	**Corrected and Scheduled for Retesting**	A defect where a corrected defect is being retested.

5.3.6 Business Impact

The Business Impact provides an estimate of the relevant effect on the business that will remain until the defect is resolved. It is optional for the Federal Student Aid Project or Program Manager to assign the following ranking to the risk of a defect with respect to its effect on the business:

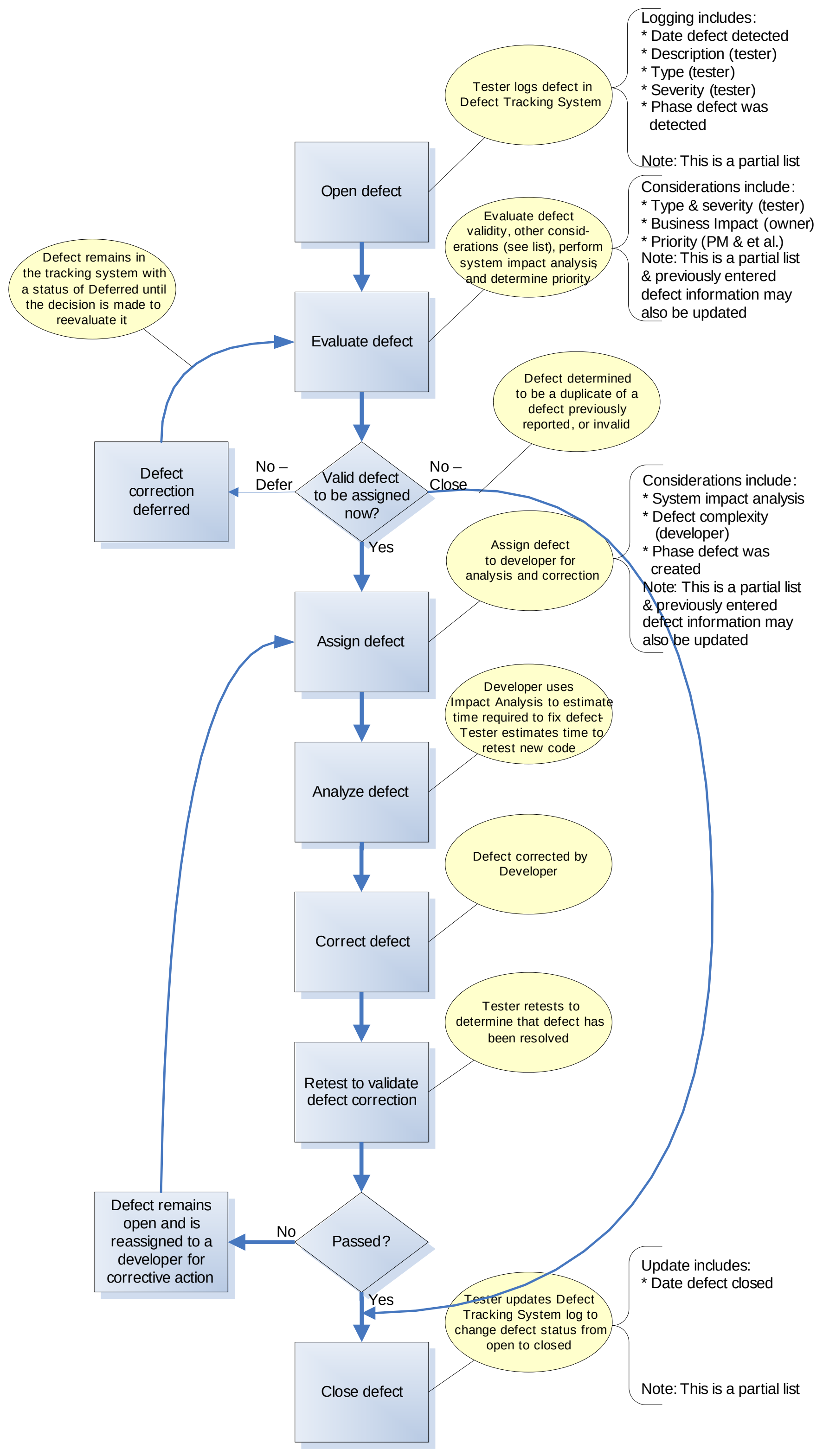
- urgent (application not available for any processing – not even for diagnostics or for viewing)
- high (unable to deliver aid (e.g., can't process applications) – The focus should be on ensuring that there is data integrity, data confidentiality, and that it won't cause harm). For example, if a stored procedure is altering financial data incorrectly, then that is mission-destructive and not mission supportive, BUT the application is still available, for processing. Also, the application is still available at least in part, for diagnostics, etc.
- medium (impedes the ability to provide aid (e.g., slowing a process or processes)
- low (additional feature/functionality that does not impact customer service).

5.4 Defect Lifecycle

Every defect has a lifecycle, beginning with entry of the defect into the defect management system and ending with defect closure. The process is as follows:

- Opening Defect
- Evaluating Defect
- Assigning Defect
- Analyzing Defect
- Correcting Defect
- Retesting to Validate Defect
- Closing Defect

Figure 5-1, Defect Life Cycle


Logging includes:
* Date defect detected
* Description (tester)
* Type (tester)
* Severity (tester)
* Phase defect was detected
Note: This is a partial list
Tester logs defect in Defect Tracking System
Open defect
Considerations include:
* Type & severity (tester)
* Business Impact (owner)
* Priority (PM & et al.)
Note: This is a partial list & previously entered defect information may also be updated
Evaluate defect validity, other considerations (see list), perform system impact analysis and determine priority
Defect remains in the tracking system with a status of Deferred until the decision is made to reevaluate it
Evaluate defect
Defect determined to be a duplicate of a defect previously reported, or invalid
Defect correction deferred
No – Defer
Valid defect to be assigned now?
No – Close
Yes
Considerations include:
* System impact analysis
* Defect complexity (developer)
* Phase defect was created
Note: This is a partial list & previously entered defect information may also be updated
Assign defect to developer for analysis and correction
Assign defect
Developer uses Impact Analysis to estimate time required to fix defect. Tester estimates time to retest new code
Analyze defect
Defect corrected by Developer
Correct defect
Tester retests to determine that defect has been resolved
Retest to validate defect correction
Defect remains open and is reassigned to a developer for corrective action
No
Passed?
Yes
Update includes:
* Date defect closed
Note: This is a partial list
Tester updates Defect Tracking System log to change defect status from open to closed
Close defect

5.4.1 Opening Defect

Once a defect is discovered, the testing team reviews the defect to ensure that it is valid. Once a defect has been determined valid, the tester that discovered the defect logs the defect in a defect management tool that captures data for all of the fields.

Table 5-5, Defect Log Required Data Elements

Defect Log Required Data Elements		
#	**Field Name**	**Field Description**
1	**Area of Introduction**	Area in which the defect was introduced into the application (e.g., requirements, design, test suite, code, etc.)
2	**Defect ID**	Unique Id for a defect (Should be generated by the tool)
3	**Defect State**	Open, Evaluated, Closed, Deferred, Assigned, Analyzed, Corrected and Not Scheduled for Retesting, and Corrected and Scheduled for Retesting
4	**Defect Type**	See **Section 5.3.1**for a complete list
5	**Date**	Date and time the defect was logged into the tracking tool
6	**Defect Synopsis**	A brief (one line) summary of the defect
7	**Defect Description**	A detailed description of the defect indicating all of the preconditions and steps needed to reproduce the defect
8	**Defect Severity**	The severity of the defect (This is entered by the tester who logs the defect)
9	**Defect Priority**	The priority of the defect (This is assigned by the project manager)
10	**Defect Complexity**	The complexity of the defect (This is entered by the developer responsible for fixing the defect)
11	**Build**	The build number of the application in which the defect was discovered
12	**Environment**	The system environment the application was operating in when the defect occurred. For example Windows, Unix etc.
13	**Platform**	The platform hosting the application was operating on when the defect occurred for example, WebSphere, WebLogic, etc. (This is applicable only for web-based applications)
14	**Due Date**	The project manager enters the due date by which the defect needs to be fixed in this field.
15	**Author**	Name of the tester that recorded the defect
16	**Responsible**	Developer responsible for fixing the defect
17	**Attachments**	Screenshots, file layouts or other relevant documentation describing the defect
18	**URL**	The URL of the web page where the defect was discovered (This is applicable only for web-based applications)

Defect Log Required Data Elements		
#	**Field Name**	**Field Description**
19	**CC**	Names of other individuals in the mailing list, for example, Author, Project Manager, Developer etc.
20	**Remarks**	Comments must include details that provide an audit trail of the problem
21	**Phase of Testing**	Integration Testing, System Testing, User Acceptance Testing, and Post Implementation Verification
22	**Testing Sub phase**	Any of the sub phases of testing described in this Handbook, for example unit component testing, unit integration testing, regression testing etc.
23	**Application Under Test**	Name of the application
24	**Browser Name & Version Number**	The name of the browser and the version number of the browser the user was running on which the defect was discovered (This is applicable only for web-based applications). May be required based on the type of application being tested.
25	**Amount of Memory**	The amount of memory (in megabytes or gigabytes) installed in the computer on which the user discovered the defect. May be required based on the type of application being tested.

5.4.2 Evaluating Defect

Once a tester has entered a defect in the defect management system, the defect is evaluated for:

- Validity
- Priority
- Business Impact

Validity includes determining that the reported error:

- Is a non-conformance of the application under test to its requirements
- Is not a duplicate of a previously reported error

If a defect is determined to be invalid (i.e., if the defect cannot be reproduced), the evaluator changes the status to rejected, and closes the defect in the Defect Management System.

When an evaluator determines that a defect is valid, the Project Manager assigns a priority and business impact to the defect.

5.4.3 Assigning Defect

Each valid defect is assigned a developer and the status of the defect is changed to **Assigned**.

5.4.4 Analyzing Defect

The developer and tester analyze the defect and if needed reproduces the defect for clarification of the issue.

Using impact analysis, the developer determines the time required to fix the defect. The testers determine the time required to retest the defect in the system based on the complexity of the test.

5.4.5 Correcting Defect

The developer determines and enters the complexity of the defect in the Defect Management Tool, corrects the defect, and changes the defect status to **Corrected**.

5.4.6 Retesting to Validate Defect and Closing Defect

The tester that logged the defect retests the defect.

- If the error is resolved, the tester changes the defect status to **Closed**.
- If the error still exists, the tester indicates the defect was not fixed and the defect is returned to the developer originally assigned to correct the problem.

5.4.7 Defect Deferral

The resolution of a defect can be deferred at the discretion of the Test Manager and the Federal Student Aid Project Manager. The defect will remain in a deferred state until reopened by the Federal Student Aid Project Manager.

5.5 Defect Management Tools

An effective defect management tool is used to record and track defects from discovery to closure and possesses the following characteristics:

- User authentication to ensure that the changes are made only by authorized users. The Test Manager will ensure that only authorized users have access to the system.
- Easy customization in order to meet the unique needs of a project.
- Effective audit trail mechanism in order to trace the origin and details of all activities.
- Advanced search capabilities.
- Email notifications controlled by user preferences.
- Scheduled reports and charts.
- Time tracking.
- Private attachments in order to add logs, screenshots, and other important details.
- Integration with Version Management System, Requirement Management System and Test Management System in order to extend traceability across the full software testing lifecycle.
- Scalability from a small project team to geographically distributed project groups, for example Federal Student Aid staff outside Washington, DC

Section 6. Test Metrics and Reporting

6.1 Overview

Test metrics indicate the efficiency and the effectiveness of testing activities, test plans, and test processes throughout all phases of testing. Metrics also identify trends, productivity, and utilization of testing resources. Effective and standardized reporting of metrics facilitates communications with stakeholders and helps with decision-making processes.

Templates have been provided in Appendix E in the forms of instructions, documents and forms. These templates should be used if the defect management tool does not provide a report to reflect the data required by Federal Student Aid.

6.2 Relevance of Test Metrics and Reporting to the Handbook Audience

The following key staff members need to fully understand and follow the requirements in the Test Metrics and Reporting section of the Handbook. Subsections following this list describe the responsibilities of each position:

- Federal Student Aid and/or Contractor Project Manager
- Federal Student Aid and/or Contractor Test Manager/Test Lead
- Federal Student Aid Enterprise Test Manager
- Federal Student Aid Application Owner
- Federal Student Aid and/or Contractor Application Development Team
- Federal Student Aid and/or Contractor Application Test Team

Table 6-1, Test Metrics Roles & Responsibilities

Title	Responsibilities
Project Manager	▪ Evaluating existing metrics ▪ Determining the appropriate metrics for a testing project, during the metric identification process ▪ Providing input into the metric reporting process ▪ Determining how to capture and create metric related information related to project status and process improvement
Test Manager	▪ Evaluating existing metrics and determining the appropriate metrics for a testing project, during the metric identification process ▪ Providing input into the metric reporting process ▪ Determining how to capture and create metric related information related to project status and process improvement

Title	Responsibilities
Federal Student Aid Enterprise Test Management	▪ Providing inputs to the defect prevention process, deliverable baselining, defect discovery process, defect resolution and process improvement processes for various management reports. ▪ Validating compliance of the testing contractor with Federal Student Aid Defect Management Requirements ▪ Analyzing metric data for trends ▪ Creating enterprise level metrics
Federal Student Aid Application Owner	▪ Participating in the metric identification process in order to evaluate existing metrics and come up with appropriate metrics for the testing project ▪ Providing input to the metric reporting process
Application Development Team	▪ Creating appropriate metrics for Unit Testing ▪ Creating appropriate test execution and defect related metrics for all the testing phases (other than unit testing)

6.3 Test Metrics Definition

Test metrics are quantitative measures of the degree to which a system, component, or process possesses a given attribute. For example, Test Efficiency is the ratio of Number of Test Suites executed, and the number of defects identified.

There are two basic types of metrics, each with a specific type of data source:

- **Base metrics**: Metrics for which data can be captured directly (Number of test suites executed, number of defects uncovered, etc.)
- **Derived metrics**: Values derived from base metrics (Test Efficiency, etc.)

6.4 Metrics Identification and Selection

The Test Manager and Project Managers are responsible for metric identification and selection. Metrics identification and selection occur during test planning, with approved metrics included in the Phase Level Test Plans. Most metrics will be derived from stakeholders with a focus on the application's formal requirements, industry practices, and lessons learned. Each metric will include what will be measured as well as the standard for acceptance and failure or rejection.

Federal Student Aid requires that all stakeholders meet before the testing process begins to identify metric recommendations that testers will use during the testing process. Discussion will also take place on how the data will be collected for metrics and how the metrics will be reported. The recommendations will first be approved by Federal Student Aid Test Manager and Project Manager, and then included in the Phase Level Test Plans. If there are no Phase Level Test Plans, metrics that will be used for the test will be defined in the master test plan.

- **Required Metrics**: Federal Student Aid requires that testing use the following sets of metrics (at a minimum):
  - Test Schedule and Effort (See **Section 6.6.1**)

    - Test Defects (see **Section 6.6.2**)
    - Test Defects by Type (see **Section 6.6.2**)
    - Defects Report and Defect Severity and Current State (see **Section 6.6.3)**
    - Test Execution (see **Section 6.6.4)**

- **Optional Metrics: Federal Student Aid** may require the following testing metrics:
    - Defect Execution Reviews (see **Section 6.6.4**)
    - Code Coverage (see **Section 6.6.5**)

The Federal Student Aid Test Manager and Project Manager must approve exclusion of any required metrics. Applications that are system critical or information sensitive must adhere to the required metrics.

    - Stakeholders may review the following types of information to assist them with making recommendations.

- **Existing Metrics**: Metrics collected during other projects or those that may be recommended by a manufacturer of a COTS application.
- **Industry Standards**: Additional metrics and defect reporting may be required based on the project type or the Federal Student Aid project manager's needs. The contractor may provide additional metrics based on the project type or best practices.
- **Lessons Learned**: Information on what did and did not work on previous test projects. This information is collected by the Quality Assurance Team and entered into the Lessons Learned Database.
- **New Metrics:** When identifying the need for a new metric stakeholders must:
    - Identify data required
    - Determine a data collection method
    - Determine a means of evaluating the data
    - Evaluate the cost/benefit of the metric

6.4.1 Choose the Appropriate Metrics

The project's metrics should be selected based on project goals, stakeholder needs, and the cost/benefit provided by each metric. In addition, ensure that the data needed to conduct the measurement can be acquired within reason, and that metrics have actual significant value to the test and development activities. The number of metrics should be kept to a minimum to ensure that the most critical information is collected, evaluated, and reported.

Accurate collection and appropriate analysis of metrics is important in ensuring that defects do not escape to production and to facilitate process improvement throughout the development life cycle. For example, a detailed analysis should be performed to determine why each issue occurred in production and not during the construction and validation phase. In addition, testing should be structured to "break the system" and not demonstrate functionality.

6.4.2 Metric Descriptions

The Test Manager may be required to provide the following information for each metric selected: name, benefit, stakeholders, description, measure procedure, measurement frequency, and reporting.

6.5 Metrics Calculation and Reporting

Metrics calculation and reporting processes include:

- Capturing and verifying data
- Analyzing and processing data
- Reporting metrics

6.5.1 Capturing and Verifying Data

Whenever possible, testers should simplify the capture and verification of test metric data by:

- Making reasonable efforts to have a testing tool perform data capture during the testing effort
- Storing all relevant testing data in a location that is easily accessible to all team members

6.5.2 Analyzing and Processing Data

The Test Manager is responsible for creating the metrics and analyzing as follows:

- Reviewing data for correctness/reasonableness
- Calculating the derived metrics based on the base metrics
- Reviewing the metrics and determining if the correct information conveys the progress of the test effort
- Automating the process of calculating the derived metrics, whenever possible and feasible

6.5.3 Reporting Metrics

Federal Student Aid requires the Test Manager to report all metrics in a consistent manner and understandable format.

Approved formats include:

- Using the metrics similar to those in **Section 6.6 Required Metrics**
- Supplementing the formats in **Section 6.6** with a metric dashboard
- Supplementing the formats in **Section 6.6** with graphs and trend lines

- Indicating acceptable results and out of range positive and negative variances using color coding

Metrics must be made available to the Federal Student Aid Enterprise Testing Group.

6.6 Required Metrics

6.6.1 Test Schedule and Effort

Test schedule and effort metrics show the scope of the planned test effort, the results of testing and the derived metrics that allow comparison of the testing effort to other testing efforts.

Required test schedule and effort metrics include the following information:

- Planned and actual test start and end dates and variances
- Number of test cases executed, passed, failed, blocked*
- Total number of test cases executed and planned
- Schedule and effort variance for each of the above elements

All of the metrics listed above are required for all projects and are to be summarized by scenario and phase when reported with "AS OF" dates included on all metrics reports. Additional metrics may be required by Federal Student Aid project management.

*Blocked cases are cases that could not be executed because of a failure that occurred in a previous test case. Blocked defect tests are tests that cannot be executed on a defect because another active defect makes the test of the original defect impossible to execute. In these cases, testing of the original defect must be effectively and explicitly deferred until the other blocking defect has been corrected.

The Test Manager will report test schedule and effort metrics using the template in Appendix E.

6.6.2 Test Defect Metrics

Test defect metrics provide information showing the extent of defects numerically and by various categories.

Required defect metrics include the following information:

- Total number of defects
- Number of defects entered
- Number of defects entered by severity and priority
- Number of defects fixed
- Number of defects rejected
- Number of defects resolved and to be verified by the testing team

The Test Manager will report test defect metrics using the Test Defect Metrics and the Defect Report matrices that follow, or a similar one that contains this data, and additional data if needed.

A Defect Trend Analysis should be done with the data to determine if the defects are increasing in subsequent phases of testing. This analysis examines the number of defects found as the testing life cycle progresses. When analyzing the number of defects found during Integration Testing versus the number found during System Testing, for example, the Test Manager could uncover a risk in the testing process and conduct corrective actions to mitigate the impact.

Templates for Test Defect Metrics and the Defect Report can be found in Appendix E.

If testing activities are performed by the contractor, the contractor's proposal must state how the test estimates are calculated.

6.6.3 Defect Severity and Current

Defect severity and current state metrics provide a snapshot of the number and types of defects at a point in time.

The Test Manager will report defect severity and current state using the template in Appendix E.

6.6.4 Test Execution

Test Execution metrics show the results of testing and derived metrics that allow comparison of the testing effort to other testing efforts:

Required test execution metrics include:

- Total Number of Test Cases or Scripts
- Total Number of Test Cases or Scripts Executed
- Total Number of Test Cases or Scripts Passed
- Total Number of Test Cases or Scripts Failed
- Total Number of Test Cases or Scripts Blocked

The Test Manager will report test execution metrics using the template in Appendix E.

6.6.5 Code Coverage Metrics

The Code Coverage metrics (which may be required for Unit Testing as determined by the Federal Student Aid architect team) shows the results of statement coverage, condition coverage, and path coverage. Required Code Coverage metrics include the following information:

- Total Number of Lines in the Code
- Total Number of Lines Unit Tested
- Total Number of Conditions in the Code
- Total Number of Conditions Unit Tested
- Total Number of Test Paths in the Code
- Total Number of Paths Unit Tested

The Development Team will report code coverage metrics using the template in Appendix E.

6.6.6 Turnover Metrics

The Contractor Turnover metric may be required to be provided to the Federal Student Aid Project Manager and Federal Student Aid Application Owner. This is a non-technical, management metric that shows the attrition rate in the contractor's testing team over a specified period.

The turnover rate is useful in evaluating the stability of the contract test team and the overall effectiveness of the testing activities. A lower rate of turnover may indicate a more effective testing process, while a higher rate of turnover can indicate a less efficient effort. While the turnover rate is not definitive in predicting testing effectiveness and efficiency, the information may help identify additional underlying issues (i.e., contractor organizational concerns or contractor Project Management problems). The Test Manager will submit staff turnover information to the Federal Student Aid Test Manager at a frequency required by the Federal Student Aid Test Manager. This information must be provided at the end of the project. The Test Manager will report staff turnover metrics using the **Tester Team Turnover Template** found in **Appendix E**.

A graphical representation of test summary information must be presented to Federal Student Aid senior management. This representation must include a summarization of the test metrics for the entire project. This information may be provided in an exportable file.

Appendix A – Acronyms and Abbreviations

Appendix A: Acronyms and Abbreviations

The Handbook uses the following acronyms and abbreviations.

Acronym	Full Title
ACS	Administrative Communication System
API	Application Programming Interface
ASD	Adaptive Software Development
ASG	Architecture Support Group
ATP	Assistive Technology Partnership
C&A	Certification & Accreditation
CC	Carbon Copy
CCB	Change Control Board
CCRB	Change Control Review Board
CFR	Code of Federal Regulations
CICS	Customer Information Control System
CIO	Chief Information Officer
CMMI	Capability Maturity Model Integration
COD	Common Origination and Disbursement
COTS	Commercial-off-the Shelf
CTE	Certified Test Engineer
CVE	Common Vulnerabilities and Exposures
DCOM	Distributed Component Object Model
DOB	Date of Birth
DSDM	Dynamic Systems Development Method
E&IT	Electronic and Information Technology
EAI	Enterprise Application Integration
ED	Department of Education
EOCM	Enterprise Operational Change Management
ETSH	Enterprise Testing Standards Handbook
FAFSA	Free Application for Federal Student Aid
FDD	Feature Driven Development
FLB	First Live Batch

Acronym	Full Title
FSA	Federal Student Aid
GUI	Graphical User Interface
HTML	HyperText Markup Language
IA	Information Assurance
ICD	Interface Control Document
ID	Identification
IEEE	Institute of Electrical and Electronics Engineers
IT	Information Technology
ITA	Integrated Technical Architecture
IVRU	Interactive Voice Response Unit
JAWS	Job Access With Speech
JMS	Java Message Service
LCM	Life Cycle Management
LOV	List of Values
MTP	Master Test Plan
OMB	Office of Management and Budget
PC	Personal Computer
PIN	Personal Identification Number
PIR	Post Implementation Review
PISP	Post Implementation Support Period
PIV	Post Implementation Verification
PRR	Production Readiness Review
RCS	Reusable Common Services
RMI	Remote Method Invocation
RTM	Requirements Traceability Matrix
SA	Security Architecture
SAIG	Student Aid Internet Gateway
SAT	System Acceptance Testing
SCR	System Change Request
SLA	Service Level Agreement
SOA	Service Oriented Architecture

Acronym	Full Title
SOAP	Simple Object Access Protocol
SOO	Statement of Objectives
SOW	Statement of Work
SSN	Social Security Number
Std	Standard
TDD	Tests Drive Design
TNR	Times New Roman
TRR	Test Readiness Review
TSV	Target State Vision
U.S.	United States
U.S.C.	United States Code
UAT	User Acceptance Tests
UDDI	Universal Description, Discovery and Integration
UI	User Interface
URL	Universal Resource Locator
VDC	Virtual Data Center
WAN	Wide Area Network
WBS	Work Breakdown Structure
WSDL	Web Services Definition Language
XML	Extensible Markup Language
XP	Extreme Programming

Appendix B – Glossary

Appendix B: Glossary

Term	Definition
Acceptance Criteria	Exit criteria that a component or system must satisfy to be accepted by a user, customer, or other authorized entity.
Accuracy	Capability of the software to provide the right or agreed results or effects with the needed degree of precision.
Actual Result	Behavior produced/observed when a component or system is tested.
Ad hoc Testing	Testing carried out informally; no formal test preparation takes place, no recognized test design technique is used, there are no expectations for results and arbitrariness guides the test execution activity.
Application	Computer software/programs designed to perform work for end users of the computer – compare with system software, including middleware and operating systems, which perform functions within the computer and support the operation of application software/programs.
Application Owner	Government representative responsible for the application being developed. The representative is responsible for project initiation, budget approval and management, deliverable signoff and confirmation of project closure. Responsibilities during the test effort may be delegated to the test manager or project manager.
Audit	Independent evaluation of software products or processes to ascertain compliance to standards, guidelines, specifications, and/or procedures based on objective criteria.
Audit Trail	Path through a sequence of events that enables reconstruction of the history of the events. Typically the path from the output of a process to the original input (e.g. data), i.e. the history of the transaction. An audit trail facilitates defect analysis and allows conduct of process audits.
Automated Testing	Testing that employs software tools that execute tests without manual intervention.
Baseline	Specification or software product that has been formally reviewed or agreed upon; that thereafter serves as the basis for further development and that can be changed only through a formal change control process.
Behavior	Response of a component or system to a set of input values and preconditions.
Best Practice	Superior method or innovative practice that contributes to the improved performance of an organization.
Black-Box Testing	Testing, either functional or non-functional, without reference to the internal structure of the component or system.
Capture/Playback Tool	Type of test execution tool where inputs are recorded during manual testing in order to generate automated test suites that can be executed later (i.e.

Term	Definition
	replayed). These tools are often used to support automated Regression Testing.
Certification	Process of confirming that a component or system person complies with its specified requirements.
Change Control Board	Stakeholders representing the interests of Federal Student Aid and the contractor reviewing change requests, making recommendations, requesting impact analyses of proposed changes, and approving change requests.
Client/Server	Network architecture in which each computer or process on the network is either a client or a server.
Code	Computer instructions and data definitions expressed in a programming language or in a form output by an assembler, compiler, or other translator.
Code Coverage	Analysis method that determines which parts of the software have been executed (covered) by the test suite and which parts have not been executed, e.g. statement coverage, decision coverage or condition coverage.
Code review and walk though	Peer review of source code intended to find and fix mistakes overlooked in the initial development phase, improving overall code quality.
Commercial Off-The-Shelf	Applications that are manufactured commercially and may then be tailored for specific use.
Compliance	Capability of the software product to adhere to standards, conventions, or regulations in laws and similar prescriptions.
Compliance Testing	Process of testing to determine the compliance of the component or system.
Component	Minimal software item that can be tested in isolation.
Component Interface Testing	Testing performed to expose defects in the interfaces and interaction between integrated components.
Component Testing	Testing of individual components of the code. These are the smallest product of code that accomplishes a specific action.
Condition	Logical expression that can be evaluated as True or False.
Condition Coverage	Percentage of condition outcomes that have been exercised by a test suite. 100% condition coverage requires each single condition in every decision statement to be tested as True and False.
Condition Outcome	Evaluation of a condition to True or False.
Configuration	Composition of a component or system as defined by the number, nature, and interconnections of its constituent parts.
Configuration Control	Element of configuration management, consisting of the evaluation, co-ordination, approval or disapproval, and implementation of changes to configuration items after formal establishment of configuration identification. [IEEE610].
Configuration Control	Group of people responsible for evaluating and approving proposed changes

Term	**Definition**
Board	to configuration items, and for ensuring implementation of approved changes. Also uses the acronym CCB.
Correctness	IEEE term for freedom from faults, meeting specified requirements, and meeting user needs and expectations.
Defect	Non-conformance of the application under test to its requirements. This includes any flaw in a component or system capable of causing the component or system to fail to perform its required function, e.g. an incorrect statement or data definition.
Defect Life Cycle	Cycle of finding a defect, reporting, assigning, coding, retesting, and closing the defect.
Defect Management	Process of recognizing, investigating, taking action and disposing of defects. Involves recording/classifying defects and identifying the impact.
Defect Management Tool	Tool that facilitates the recording and status tracking of defects.
Defect Priority	Process of assigning the priority in which a defect should be fixed.
Defect Report	Document that reports any flaw in a component or system that can cause the component or system to fail to perform its required function.
Defect Tracking Tool	See *Defect Management Tool*.
Deliverable	Work product that must be delivered to someone other than the (work) product's author based on contractual guidelines.
Development Team	Responsible for designing and performing unit testing and supporting the subsequent testing efforts by analyzing change requests and updating code as required.
Development Testing	Formal or informal testing conducted during the implementation of a component or system, usually in the development environment by developers.
Documentation Testing	Testing the quality of the documentation, e.g. user guide or installation guide.
Enterprise Test Management	Responsible for the definition and compliance of organization wide testing standards and providing leadership and direction to the Test manager and Test team.
Entrance criteria	Set of conditions for permitting a process to go forward with a defined task, e.g. test phase.
Exit Criteria	Set of conditions agreed upon with the stakeholders, for permitting a process to be officially completed.
Expected Result	Behavior predicted by the specification, or another source, of the component or system under specified conditions.
Fail	Actual result does not match expected result.
Failure	Deviation of the component or system from its expected delivery, service, or result.

Term	Definition
Feature	Attribute of a component or system specified or implied by requirements documentation (e.g., reliability, usability, or design constraints).
First Live Batch	Period of time after the system enters production where defects are monitored closely and fixed via emergency fixes, etc. Also known as the stabilization period.
First Live Batch Testing	Validation of critical functions of applications once the software is installed into production.
Formal Review	Review characterized by documented procedures and requirements, e.g. inspection.
Functional Requirement	Requirement that specifies a function that a component or system must perform.
Functional Testing	Subset of system testing. Testing based on an analysis of the requirements of a component or a system.
Functionality	Capability of the software product to provide functions meeting stated and implied needs when the software is used under specified conditions.
Impact Analysis	Assessment of change to the layers of development documentation, test documentation, and components, in order to implement a given change to specified requirements.
Integration	Process of combining components or systems into larger assemblies.
Integration Testing	Phase of software testing in which individual software modules or units are combined and tested as a group.
Interface Control Document	Inputs and outputs of a single system or the interface between two systems or subsystems. The Interface Control Document (ICD) is used as a communication tool for test teams.
Intersystem Testing	Integration test type that is concerned with testing the interfaces between systems.
Maintenance	Modification of a software product after delivery to correct defects, to improve performance or other attributes, or to adapt the product to a modified environment.
Management Review	Systematic evaluation of software acquisition, supply, development, operation, or maintenance performed by or on behalf of management. Monitors progress, determines the status of plans and schedules, confirms requirements and the system allocation, and evaluates the effectiveness of management approaches to achieve fitness for purpose.
Master Test Plan	Test plan that typically addresses multiple test levels. See also *Test Plan*.
Metric	Measurement scale and the method used for measurement.
Module	See *Component*
Pass	Actual result matches its expected result.

Term	**Definition**
Pass/Fail Criteria	Decision rules used to determine whether a test item (function) or feature has passed or failed a test.
Peer Review	Review of a software work product to identify defects and improvements.
Performance	Degree to which a system or component accomplishes its designated functions within given constraints in regards to processing time and throughput rate.
Performance Indicator	High-level metric of effectiveness and/or efficiency used to guide and control progressive development, e.g. lead-time slip for software development.
Performance Testing	Process of testing the degree to which a system or a component accomplishes its designated functions within given constraints in regards to processing time and data transfer rates.
Performance Testing Tool	A tool to support performance testing that usually has two main facilities: load generation and test transaction measurement. Performance testing tools normally provide reports based on test logs and graphs of load against response times.
Phase Level Test Plan	Test plan that typically addresses a single test phase. See also *Test Plan*.
Post Implementation Support Testing	Post Implementation Support Period is the phase following the initial implementation of the application into the production environment during which the development contractor is responsible for defect resolution.
Post Implementation Verification	Post Implementation Verification is the final phase of testing that occurs after the application is deployed in production.
Production Environment	Hardware and software products installed at users' or customers' sites where the component or system under test will be used. The software may include operating systems, database management systems, and other applications.
Production Readiness Review	Final, formal, and documented decision point before a new application or a significant release of an existing application enters Federal Student Aid's production environment and is exposed to end-users.
Project	Unique set of coordinated and controlled activities with start and finish dates undertaken to achieve an objective conforming to specific requirements.
Project Manager	Responsible for the overall success of a development or maintenance project.
Program	See *Application*
Quality	Degree to which a component, system or process meets specified requirements and/or customer needs and expectations.
Regression Testing	Testing of a previously tested program following modification to ensure that defects have not been introduced or uncovered in unchanged areas of the software.
Requirement	Condition or capability needed by a user to solve a problem or achieve an objective that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed

Term	Definition
	document.
Requirements Liaison	Responsible for facilitating communication between the project's Requirement Manager and Test Manager.
Re-Testing	Re-running test cases that failed in order to verify the success of corrective actions.
Risk Identification	Process of identifying risks using techniques such as brainstorming, checklists, and failure history.
Risk Management	Systematic application of procedures and practices to the tasks of identifying, analyzing, prioritizing, and controlling risk.
Security	Attributes of software products that prevent unauthorized access, whether accidental or deliberate, to programs and data.
Security Testing	Testing to determine the security of the software product. See also *Functionality Testing*.
Security Testing Tool	Tool that provides support for testing security characteristics and vulnerabilities.
Service-Oriented Architecture	Architecture that relies on service orientation as its fundamental design principle. Service-orientation describes an architecture that uses loosely coupled services to support the requirements of business processes and users.
Software	Computer programs, procedures, and associated documentation and data, pertaining to the operation of a computer system – see also *Application.*
Specification	Document that specifies, in a complete, precise and verifiable manner, the requirements, design, behavior, or other characteristics of a component or system, and, the procedures for determining whether these provisions have been satisfied.
Subsystem Testing	Testing of integrated components that form a major part of the whole system.
System	Collection of components organized to accomplish a specific function or set of functions.
System Testing	Testing that attempts to discover defects that are properties of the entire system rather than of individual components.
Test	Demonstration of how well an application functions based on business objectives and requirements.
Test Approach	Implementation of the test strategy for a specific project.
Test Automation	Use of automated software to perform or support test activities, e.g. test management, test design, test execution, and results checking.
Test Case	Set of input values, execution preconditions, expected results and execution post conditions developed for a particular objective or test condition, such as to exercise a particular program path or to verify compliance with a specific requirement.

Term	Definition
Test Case Management Tool	Tool for management of test cases for software and hardware testing.
Test Condition	Item or event of a component or system that could be verified by one or more test suites, e.g., a function, transaction, feature, quality attribute, or structural element.
Test Cycle	Execution of the test process against a single identifiable release of the test object.
Test Data	Data that exists before a test is executed, and that affects or is affected by the component or system under test.
Test Environment	Environment containing hardware, instrumentation, simulators, software tools, and other support elements needed to conduct a test.
Test Execution	Process of running a test on the component or system under test, producing actual result(s).
Test Execution Automation	Use of automated software, e.g. capture/playback tools, to control the execution of tests, the comparison of actual results to expected results, the setting up of test preconditions, and other test control and reporting functions.
Test Execution Phase	Period of time in a software development life cycle during which the software product is evaluated to determine whether or not requirements have been satisfied.
Test Input	Data received from an external source by the test object during test execution.
Test Item	Individual element to be tested.
Test Lead	Responsible for creating, tracking and maintaining phase level test plans, assigning tasks to the test team and reviewing deliverables.
Test Level	Group of test activities that are organized and managed together. Examples of test levels are component test, integration test, system test, and acceptance test.
Test Log	Chronological record of relevant details about the execution of tests.
Test Manager	Responsible for project management of testing activities and resources and evaluation of a test object.
Test Management	Planning, estimating, monitoring and control of test activities, typically carried out by a Test Manager.
Test Management Tool	Tool that provides support to the test management process. Contains capabilities, such as test ware management, scheduling tests, logging results, progress tracking, incident management, and test reporting.
Test Phase	Distinct set of test activities collected into a manageable phase of a project, e.g. the execution activities of a test level.
Test Plan	Document describing the scope, approach, resources, and schedule of intended test activities.

Term	**Definition**
Test Procedure	Document specifying a sequence of actions for the execution of a test. Also known as test suite or manual test script.
Test Process	Fundamental test process comprised of planning, specification, execution, recording, checking for completion and test closure activities.
Test Readiness Review	A multi-disciplined technical review to ensure that the subsystem or system under review is ready to proceed into formal test. The Test Readiness Review assesses test objectives, test methods, and procedures, scope of tests, traceability of planned tests to program requirements and user needs and safety and confirms that required test resources have been properly identified and coordinated to support planned tests. The Test Readiness Review determines the completeness of test procedures and the compliance with test plans and descriptions.
Test Scenario	See *Test Procedure Specification.*
Test Suite	Refers to a test procedure specification, particularly an automated one.
Test Specification	Document that consists of a test design specification, test case specification and/or test procedure specification.
Test Strategy	High-level description of test levels to be performed and testing within those levels for an organization or program (one or more projects).
Test Suite	Set of several test scenarios, test cases, test procedures and test scripts for a component or system under test.
Test Summary Report	Document summarizing testing activities and results. Contains an evaluation of the corresponding test items against exit criteria.
Test Type	Group of test activities aimed at testing a component or system focused on a specific test objective.
Tester	Involved in the testing of a component or system.
Testing	Process consisting of all life cycle activities, both static and dynamic, concerned with planning, preparation and evaluation of software products and related work products to determine that requirements are satisfied, to demonstrate that the application is fit for purpose and to detect defects.
Traceability	Ability to identify related items in documentation and software such as requirements with associated tests.
Unit Testing	Test used to validate that individual units of source code are working properly. A unit is the smallest testable part of an application.
Unit Integration Testing	Testing two or more software units that have been integrated to ensure that it works together as intended.
Usability Testing	Measuring how well people can use some human-made object (such as a web page, or a computer interface) for its intended purpose.
User Acceptance Testing	Formal testing with respect to Application Owner needs, requirements, and processes conducted to determine whether a system satisfies the acceptance

Term	Definition
	criteria and to enable the user, customers, or other authorized entity to determine whether to accept the system.
Web-Based Architecture	Composed of services and technologies that enable applications to function in an internet and/or intranet environment through a web browser user interface.
White Box Testing	Testing the internal structure of a component or a system with access to source code and configuration parameters of components. Also known as clear box testing, glass box testing, or structural testing.

Appendix C – Cross Reference for Intended Audience to Relevant Handbook Sections

Appendix C: Cross Reference for Intended Audience to Relevant Handbook Sections

Table C-1, Cross Reference for Intended Audiences identifies relevant sections of the Handbook for a variety of types of roles, based on responsibilities. Federal Student Aid requires all of these participants to learn and comply with the processes identified with roles in the matrix. Everyone is also encouraged to become familiar with the rest of the Handbook.

Table C-1, Cross Reference for Intended Audience

Roles	Section						Appendix
	1	2	3	4	5	6	D
Federal Student Aid							
Application Owners	✓	✓	✓				
Project Manager and Program Manager	✓	✓	✓	✓	✓	✓	
Enterprise Test Manager	✓	✓	✓	✓	✓	✓	✓
Test Manager	✓	✓	✓	✓	✓	✓	✓
Development Manager (small apps)	✓		✓	✓	✓	✓	✓
Development Team (small apps)			✓	✓	✓	✓	✓
Requirement Team	✓		✓		✓	✓	
Test Suite Reviewer			✓		✓	✓	
Test Leads (each test phase as assigned)	✓		✓	✓	✓	✓	✓
Integration Testers			✓	✓	✓	✓	✓
UAT Tester			✓	✓	✓	✓	✓
PIV Testers			✓	✓	✓	✓	✓
Contractor							
Project Manager	✓	✓	✓		✓	✓	
Development Manager	✓	✓	✓	✓	✓	✓	✓
Test Manager	✓	✓	✓	✓	✓	✓	✓
Requirements Manager	✓	✓	✓		✓	✓	
Development Team			✓	✓	✓	✓	✓
Requirement Team			✓			✓	
Test Leads (all test phases)	✓		✓	✓	✓	✓	✓
Integration Testers			✓	✓	✓	✓	✓

Roles	Section						Appendix
	1	2	3	4	5	6	D
System Testers			✓	✓	✓	✓	✓
UAT Support Team			✓	✓	✓	✓	✓
PIV Testers			✓	✓	✓	✓	✓

Appendix D – Testing Techniques

Appendix D: Testing Techniques

Overview

To the untrained observer, software testing may appear to be intuitive, involving little more than running a program several times to see whether it meets requirements. In fact, software testing techniques are largely systematic, beginning with an analysis of the requirements before any code is written, and involving the need for testers to thoroughly analyze the system under development using various methods and tools. The following subsections provide an introduction and overview of the most common testing techniques.

Test Managers, Test Leads, Developers, and Test Engineers should utilize these techniques in their testing processes to obtain what the IEEE Standard Glossary of Software Engineering Terminology defines as "*Correctness*" or freedom from faults, meeting specified requirements, and meeting user needs and expectations. Using these techniques, along with others that are not listed but may be appropriate, will ensure a high level of correctness in Federal Student Aid software.

Specialized Testing Techniques

Effective testing is critical to the success of every application development or enhancement effort. In addition to core application functionality, effective testing also includes the use of solid testing methodology and appropriate testing techniques on specialized areas related to the application (for example, Graphical User Interface (GUI) testing, Web based testing, 508 Compliance Testing, Commercial-off-the-Shelf (COTS) Products Testing, Evergreen Testing, testing of specific architectures, or multiple environments and Security Testing). In each case, finding, logging, and classifying bugs and defects by type are required to facilitate prompt correction.

1. EAI/ITA/SA Guidance

The test planning and test phases sections of the Handbook apply to EAI/ITA/SA testing. The following sections provide a brief overview and examples of EAI, ITA and SA testing.

EAI TESTING

Enterprise Application Integration (EAI) tests the validation of the messaging infrastructure and integration capability. The main goal is to standardize interfaces to new and legacy systems in support of Federal Student Aid's Target State Vision (TSV).

EAI performs the appropriate application development tests on adapters and software components the ITA team has written (Application Programming Interfaces, Dynamic Link Libraries, etc.), using test procedures developed by ITA. EAI tests COTS components that support their mission prior to installing those components in the production environment (e.g., WebSphere, MQ (MQSeries and upgrades from IBM). In these cases, EAI uses test procedures developed by the VDC to verify that the infrastructure created is acceptable to perform COTS testing and is functioning properly, before and after installation of the COTS components. EAI supports developers by creating the middleware infrastructure in a development environment to allow developers to operate in environments that mimic the production environment where the software will eventually operate.

Example - EAI Testing

An example of EAI Testing is testing the messaging capabilities of IBM WebSphere MQ during batch and real time transaction processing.

Impact of EAI Testing on Test Planning

During Test Planning, EAI works with ITA and the application teams to plan application tests and to create the required test suites. EAI and ITA create test environments to support the application developers. This environment helps to identify infrastructure issues and provides collaborative assistance with developers during the test effort.

Test Planning efforts conform to the procedures in Section 3 of the Handbook. If steps are excluded, the Federal Student Aid Project Manager or Federal Student Aid Test Manager must approve the exclusions. Key planning considerations include selecting applications that could be affected by the updated EAI source code and the messaging infrastructure and appropriate mapping of the infrastructure to test activities. Test planning needs to define who is responsible for obtaining the test plans and test scripts from the application teams, updating them and executing the test suites and test scripts. Whenever there is a change in the functional requirements of an application, the appropriate test plans and test scripts must be updated. The plan should also address how the test results and metrics will be communicated to the Project Manager and Test Manager who will be included in the plan.

Impact of EAI Testing on Test Phases:

Unit Testing

Unit testing is performed whenever there is a modification in either the EAI source code or the MQ Series custom code. The procedure for Unit Testing is described in Section 4.2.

Integration Testing

Integration Testing is not applicable to EAI Testing.

System Testing

System Testing involves Intersystem Testing, Regression Testing, and Performance Testing. The procedure for System Testing is described in Section 4.4. The key element of System Testing is to validate whether the updated messaging infrastructure and/or EAI Source code integrates with the overall application infrastructure.

During System Testing, test scripts are created, automated testing is performed using E-metRx, and individual test results are documented. These scripts are stored in libraries and can be accessed at later date to carry out regression testing on the applications as changes take place in the middleware infrastructure and RCS components. For EAI to be effective in system testing, the recommendation is for the application team to store updated test plans, test suites and version/release information in an enterprise repository that is accessible to all teams as well as the VDC and ITA. It is also recommended that the application teams store other associated application artifacts, such as application architecture diagrams, requirements, and PRR documents.

Each test suite will be executed as a standalone unit, and the results will be documented in a Test Run Table in the format shown below. Output files, where indicated, will be checked for status. If the status is "true" and all observed results meet the expected results, the test will be deemed a success (Pass). If a "false" status is found, one or more issues will be logged and submitted for resolution with the test deemed not successful (Fail). When modifications to correct the problem(s) have been applied to the test environment, the failed test suite will be rerun. This process will be repeated as necessary until all exceptions are satisfactorily resolved. Specifically, when a failure is encountered, the issue will be identified, logged, and corrected, and the test suite will be run in its entirety from start to finish. If a test does not produce the "Expected Result(s)," then the test will be deemed a "Fail." If a test suite does produce the "Expected Result(s)," then the test will be deemed a "Pass."

Table D-1, Test Run Table

Test suite #				
Step #	**Date**	**Exp. Results = Actual Results**	**Pass / Fail**	**If Fail, enter Issue #**

UAT

UAT involves Regression Testing of the affected applications to test updates in the messaging infrastructure and the EAI source code. The procedure for UAT is described in **Section 4.5**. UAT involves test execution of test suites written for infrastructure and service orchestration, migration, mediation, governance, and security. UAT includes verification, management and monitoring requirements, inspection of documents, logs and dash boarding components, verification of training and deliverable requirements and conducting a training session.

Post Implementation Verification

Post Implementation Verification is carried out as shown in Section 4.6 of this Handbook.

ITA TESTING

Integrated Technical Architecture (ITA) testing validates the integrated, enterprise wide technical architecture on which the Federal Student Aid applications are deployed. The testing mainly involves validating the COTS products, Architectural Upgrades, and RCS Development and Support.
ITA creates and maintains a series of RCS components that provide connectivity between individual applications, and software appliances such as Google's search engine. Federal Student Aid has acquired these appliances to avoid recreating functionality already available in COTS products. ITA currently performs four types of tests as listed below:

1. **RCS –** Testing to support the development and maintenance of components that provide interfaces with appliances that are integrated with Federal Student Aid applications (testing is similar to testing during application development and maintenance).

2. **Evergreening –** Testing of COTS product upgrades during integration into the Federal Student Aid environment. Regression testing is conducted to ensure that introduction of the new COTS product or upgrade doesn't break the existing infrastructure.
3. **Integration –** Installation of new or updated applications into the production environment (similar to Evergreening), integration testing is conducted to ensure that the new or updated application doesn't break the existing infrastructure, but this testing ensures that the application functions correctly within the infrastructure.
4. **Password changes –** Federal Student Aid policy requires changing applications' database passwords every 60 to 90 days. ITA performs limited regression testing following these changes to ensure that applications continue to function correctly.

Example - ITA Testing

An example of ITA Testing is to test the advance search capabilities of the Google appliances installed in the ITA environment.

Impact of ITA Testing on Test Planning:

Test Planning efforts must be followed in the same manner as other test efforts and must conform to the procedures in **Section 3** of the Handbook. If steps are excluded, the Federal Student Aid Project Manager or Test Manager must agree to the steps. Test planning needs to take into account who will execute the test suites and test scripts (the Application Test Team (Federal Student Aid Application Owners) or the ITA Team) and how the test results and metrics will be communicated to the Project Managers and Test Managers.

The functional and non-functional requirements for COTS products upgrade testing are defined using expert judgment and vendor support documentation. Since each product upgrade affects specific functionality, the scope of testing requirements is defined by the scope of the changes in the upgrade.

Based on input from vendor documentation, vendor support, and internal team discussions, ITA identifies the critical tests in order to ensure the success of the infrastructure component upgrade.

As part of ensuring communication of enterprise wide changes to infrastructure components, the EOCM process monitors all major changes. ITA provides testing schedules, and the EOCM committee oversees the coordination of the change process. For smaller changes, the team that requests and/or performs the change monitors the testing.

Impact of ITA Testing on Test Phases:

Unit Testing

Unit testing is carried out whenever there is a modification to either the RCS code or the Custom Code of the COTS products. The procedure for Unit Testing has been described in Section 4.2.

Integration Testing

Integration Testing is not conducted in ITA Testing.

System Testing

System Testing involves Intersystem Testing, Regression Testing, and Performance Testing. The procedure for System Testing has been described in **Section 4.4**. The key element of System Testing is to validate whether the updated architecture, RCS Code, and/or the COTS products integrate well with the overall application infrastructure.

ITA uses internally developed procedures for testing infrastructure component upgrades, tailored for the specific environment where the upgrade is performed. The baseline for developing the test procedures is the vendor support documentation (release notes, installation and administration guides, best practices).

For each testing activity, ITA records the outcome either in a test matrix (developed internally) or in the change request record.

Currently, the ITA tech lead is responsible for validating the testing procedures, Test Readiness Review, and testing activities to be performed. The ITA Test Team performs the testing for each upgraded component.

UAT

UAT involves Regression Testing of the affected applications as a result of updates in the Architecture, RCS, and/or COTS products. The procedure for UAT is described in **Section 4.5**.

The testing requirements in this category are defined by the application teams and incorporated in the documented testing processes. ITA provides test requirements input, based on expert judgment, when requested by the application teams.

Post Implementation Verification

Post Implementation verification is conducted as shown in **Section 4.6** of the Handbook.

SA TESTING

Security Architecture (SA) testing involves validation of access controls, provisioning, de-provisioning, self-registration, delegated administration, and simplified sign-on across the Federal Student Aid enterprise. The architecture includes access management tools, identity management tools, enterprise policy repositories, enterprise user repositories, and other related security components.

SA tests fall into one of three categories:

1. In an SA only change, a checklist is used to evaluate the change within the security architecture. SA then invites all of the application groups to test applications in the new environment.
2. For an application only modification, SA performs an application specific review of the modified application to ensure that the application still works within the SA environment.
3. For Integration related changes, SA performs a high-level review to ensure that other applications continue to work with the new application.

Note: *Integration related changes occur the first time an application and the SA environment are integrated. When this occurs, SA invites the other production applications to retest SA related functionality to determine whether the integration of the new application broke any existing functionality.*

As shown below, the Security Architecture Test/Certification Matrix document provides a checklist for testing changes and certifying the integrity of the Security Architecture Production environment. The matrix is used to ensure that the functionality and operability of the Security Architecture environment has not been adversely affected by configuration changes, patch installation, or any other changes. The tester should follow the test matrix and provide answers along with comments when appropriate. The current process ends when the Go/No Go decision is made for each change.

Table D-2, Security Architecture Test/Certification Matrix

Critical or Non-Critical Item	Test/Verification Item	Results (Yes/No)	Item Verified by	Comments

Example - SA Testing

An example of SA Testing is to check the password parameters enforced by Tivoli Access Manager by validating the access rights of appropriate users.

Impact on Test Planning:

Test Planning efforts must be followed as they are in other test efforts and must conform to the procedures in **Section 3** of the Handbook. If steps are excluded, the Federal Student Aid Project Manager or Test Manager must agree to those steps. The test planning needs to take into account how the test results and metrics will be communicated to the Project Managers and Test Managers.

Testing requirements are driven by the upgrade functionality. The SA Test matrix lists the functional requirements and non-functional requirements for each COTS Upgrade. The SA team manages the test effort. In order to mitigate risks, the SA Team and SA integrated application representatives are involved in testing. The SA Team and integrated application Federal Student Aid Application Owners decide what will be tested and how to measure success or failure.

Impact on Test Phases:

Unit Testing

Unit Testing is not conducted in SA Testing.

Integration Testing

Integration Testing is not conducted in SA Testing.

System Testing

System Testing involves Intersystem Testing, Regression Testing, and Performance testing. The procedure for System Testing has been described in **Section 4.4**. The key element of System Testing is to validate whether the security architecture integrates well with the overall application infrastructure. The Test/Certification Matrix is used for System Testing.

UAT

UAT involves Regression Testing of the affected applications as a result of the updates in the Security Architecture. The procedure for UAT has been described in **Section 4.5**. The Test/Certification Matrix is used for UAT.

Post Implementation Verification

Post Implementation Verification is conducted as shown in **Section 4.6** of the Handbook. The Test/Certification Matrix is used for Post Implementation Verification.

2. Infrastructure Testing

The level of testing required for all infrastructure updates (e.g., Google search engine, system password changes, IP address changes) should be determined during the Impact Analysis process (standard change management process of EOCM). Analysis of test results feeds into all go/no-go decisions.

Required Infrastructure Testing includes the following:

- Network/Firewall Updates
  - Testing is mandatory for all mission critical and mission important applications / those sending out data.
- Domain Name System (DNS) (Example: Mail/Relay Services)
  - Testing will be mandatory for those projects specifically impacted by the DNS change or the new DNS entries.
- Server Patches (Example: Daylight Savings Time)
  - Testing by system team based on the complexity and the risks related to the update.
  - All systems impacted by the change must test.
- New Servers (Includes New Databases and new Custom Systems, e.g., IPM)
  - Testing of configurations (capacity/settings) of servers is required.
  - The server must be tested for stability, functionality and security.
  - All new servers must have security scans to point out security vulnerabilities (based on security requirements).
- COTS Upgrade (Examples: TIM/TAM or Informatica)
  - Testing by application owners is required if the product is used by the application.
  - Integration Testing is highly recommended.
  - Security Scans prior to production are required to point out security vulnerabilities.
- System Maintenance Releases (Code Promotion/Install to Production; Example: XML Registration)

    - Security testing is part of C&A requirements.
    - EAI (Enterprise Application Integration) will be required to do rudimentary testing.
    - Deployment/installation testing will be performed as part of Post Implementation Verification.
- System Level Database Changes
    - Testing by the application owner is highly recommended.
    - Security scans must be performed to identify potential security vulnerabilities.
    - Validation by systems team mandatory for all mission critical and mission important applications.
    - System component testing of database changes will follow the standards set in this handbook.
- System password Changes by data center (Example: Oracle Database Password)
    - Testing by the systems team is mandatory. This is to ensure access is not in jeopardy.

3. Service Oriented Architecture Testing

Service Oriented Architecture (SOA) is an approach to constructing enterprise systems from a set of collaborating services which, when combined, are comprised of business use cases. The services are self-describing and can be assembled ad hoc to comprise business processes.

Web services are a common platform that enables SOA. There are three main components to web services:

1. **Simple Object Access Protocol (SOAP):** Provides the envelope for sending messages to web services
2. **Web Services Definition Language (WSDL):** Forms the basis for web services
    - A service provider describes its service using WSDL
3. **Universal Description, Discovery and Integration (UDDI):** Registries can be searched to quickly, easily, and dynamically find and use web services

Note: *The services used in SOA are not limited to web services, but can include other technologies such as Distributed Component Object Model (DCOM) and XML over Remote Method Invocation (RMI).*

The following are key best practices when testing SOA and, in particular, testing web services:

- Unit and Functional Testing should occur as early as possible in the development cycle and should provide a basis for developing regression test cases
    - Regression test cases must be used frequently to ensure the robustness of each service.
- Test suite management and setup is paramount to reducing the time required for setting up and maintaining functional and regression tests.

- Processing of services can involve complex interaction of legacy and third party systems.
  - End-to-end visibility of each component involved in the processing of a specific service to identify and resolve potential issues must occur.
  - Test Harness – is frequently a good way to "institutionalize" exposing functionality of the system (e.g., via Web Services) for coarse-grain functionality typical for acceptance tests and major system scenarios (and even external invocation). Wiring up a test harness enables ease of extending the breadth of the tests as new functionality is added. You can build test harnesses that are very automated in terms of capturing the user input (using a record mode) and capturing the output, and getting a user's (in this case typically a tester) acknowledgement that the output is correct. Each captured test suite is added to a test database for later playback orchestration.
- Test coverage in SOA must consider how each element operates within a larger process, not just how an element operates in isolation.
  - SOA testing best practices need to assign risk weights to each element as part of the larger overall process.
  - When testing SOA, understanding how each service functions within dependent processes and how the dependent processes function end-to-end, is critical to determining proper test coverage.
- Testing and de-bugging issues in SOA requires a team effort that provides visibility into the entire path of the service.
  - Identifying proper test coverage and the ability to diagnose and fix issues will often involve not only the tester, but operations experts, developers, process matter experts, and business analysts.
- The process for testing best practices for SOA should be as follows:
  - Component interfaces should be tested as they are when added to the overall integration architecture.
  - If the process requires systems to be available for complete end-to-end testing that are unavailable or not built, this problem can be reduced significantly by simulating missing components in the testing process.
- Testing of SOA should focus on the following areas:
  - **Functionality**: The service must return the expected response for all possible input values.
  - **Performance**: The response time must be acceptable for a single instance of the service call and under anticipated load conditions. The test must check for scenarios that may cause performance degradation.

- **Backward Compatibility**: As business evolves, new consumers of the service may force changes in the input and output values. These changes must not compromise functionality for the original consumers.
- **Inter-operability**: The service must maintain correct functionality on all platforms and with all communication protocols intended for use. Proper inter-operability testing requires a test environment that simulates as closely as possible the real world production environment.
- **Compliance**: Each service must adhere to governing standards to maintain integrity and robustness. A compliance-testing tool can be used to verify compliance of each service during development.
- **Security**: Each service must comply with security constraints. Web services exposed to the Internet can pose a significant security risk.

Example - SOA Test

An example of a service developed for Federal Student Aid is the PIN web service. This service provides validation that a specific combination of SSN, first two letters of a last name, DOB, and PIN exists in the PIN database. The service returns either a correct response if the record exists or an incorrect response if the record does not exist. Multiple applications may use the service, and this should be considered when planning the testing.

Impact on Test Planning

Planning for SOA testing differs from traditional enterprise application test planning, primarily in the area of project test strategy. SOA testing adds considerable complexity to and emphasis on the following areas:

- **Test Coverage**: Testing must encompass how each service operates within the broader business processes.
  - Since there will likely be many different business processes utilizing each service, process visibility is critical to defining process coverage.
  - Traditional application testing assures that the user interface performs as expected. SOA testing needs to go behind the user interface to expose the entire process and the data driven through the process.
- **Test Tools**: Test automation plays a more significant role in SOA Testing because of the necessity to run Functional, Regression, and Performance Testing more frequently within the development lifecycle.
  - Testing tools must be able to test other technologies that can expose services, such as Java Message Service (JMS), which is a middleware; Application Programming Interface (API) for sending messages between two or more clients; and RMI, which is an API for performing remote procedure calls.

- **Training Requirements**: SOA testing may require more specialized skills in areas such as writing custom interfaces to access services or to simulate missing components required to run services (e.g., simulate a back-end or third party component).
- **Level of Regression Testing**: For SOA, regression testing typically occurs more frequently due to changes or additions in dependent services and processes.
  - Because services are shared resources, a change to one service can have potential impacts on many processes.

Impact on Testing Phases

SOA testing encompasses the same testing phases as traditional enterprise applications; however, emphasis is placed on the following:

- Unit and Functional testing should occur as early as possible in the development cycle and should provide a basis for developing regression test suites.
- Within the Functional Testing phase, the ability to perform asynchronous testing is often required.
  - With SOA, services are often not synchronized so at times there may be a need to test business processes out of sequence.
  - The test team needs the ability to address asynchronous communications mechanisms such as the polling of data, polling-through-proxy, and callbacks through WS-Addressing as well as other standards.
- More frequent Regression Testing is typically necessary to cope with potential impacts to changes in services shared by many different processes.
- Network sensitivity testing is more crucial for SOA because components typically are distributed over a Wide Area Network (WAN), which can impose significant performance limitations.
- Within all testing phases, it is important to analyze and understand the impact of changes.
  - One of the major challenges in SOA testing is the decision of how to effectively test changes that occur.
  - A determination must be made as to which changes potentially cause major risks and where the best places are for the test team to invest efforts, as well as where minimal validation is needed to optimize resources.
  - In traditional application testing processes, the testing challenge has been to understand the changes development made and decide whether testing is needed.
  - Understanding the impact of changes is more significant in shared services due to the number of services involved, the number of internal dependencies, and the number of applications using shared services.

4. Evergreen Testing

Federal Student Aid needs to remain up to date with newer versions of COTS software. To facilitate these updates, Federal Student Aid has developed a policy called Evergreen Testing (or Evergreening) for keeping software and hardware resources up to date. The configuration and other changes to the infrastructure and COTS products should be published periodically.

For example, a simple Evergreening policy might involve:

- Replacing all computers older than four years with new models
- Acquiring the most recent operating system release
- Updating the operating system software to a newer version using the existing hardware

Evergreening policy should state whether the purpose is:

- Replacing technology to maintain an acceptable level of reliability
- Acquiring improved speed and capacity at an acceptable price
- A combination of the two

Whatever the purpose, the Evergreening process is vital to ensure that Federal Student Aid systems implement the latest version. By implementing an effective Evergreen Testing policy, Federal Student Aid manages the risks associated with the integration of a new version of COTS software.

Example - Evergreen Test

An update to a newer version of COTS software, or the testing of an application that has been integrated with a newer version of WebSphere.

Test Planning

Test Planning must conform to the procedures in **Section 3** of the Handbook.

Test Phases

- **Unit Testing:** does not require Evergreen Testing.
- **Integration Testing**: does not require Evergreen Testing.
- **System Testing**: may require Evergreen Testing as shown in **Section 4.4** of the Handbook.
- **UAT**: may require Evergreen Testing as shown in **Section 4.5** of the Handbook.
- **Post Implementation Verification**: may require Evergreen Testing as shown in **Section 4.6** of the Handbook.

5. Graphical User Interface (GUI) Testing

GUI Testing, sometimes referred to as User Interface (UI) testing, involves testing the interaction between the software and the user. This includes how the application handles keyboard and mouse input and how the interface displays screen text, images, buttons, menus, dialog boxes, icons, toolbars, and more.

Before GUI testing can take place, the GUI and underlying functionality must exist. Initially, testers normally perform functional GUI testing manually. As the GUI becomes more stable, testing may move into a more automated process where faster Regression Testing can be implemented to validate that functionality has not been broken due to subsequent software changes.

Correctness is important for software in general; however, it is particularly important for user interface software and GUI Testing. The GUI interface represents the aspect of the software that is directly visible to users. If the user interface is incorrect, the user perception will be that the software is incorrect, regardless of the correctness of the underlying functionality. Software correctness includes:

1. **Verification**: Verification failure results in software containing potential faults or flaws.
   - Testers need an effective understanding of the purpose and functionality intended for the interface, and testing must validate that all of the interfaces function as designed.
   - Testers testing the GUI need to understand that the behavior of the GUI is extremely important.
2. **Validation**: Questions that need to be asked include:
   - Can windows be resized, moved, and scrolled?
   - Does the window properly regenerate when overwritten and then recalled?
   - Are all relevant pull-down menus, tool bars, scroll bars, dialog boxes, buttons, icons, and other controls available and properly displayed?
   - Is the name of the window properly represented?
   - Do multiple or incorrect mouse clicks within the window cause unexpected side effects?
   - Does the window close properly?
   - Are consistent color schemes used?
   - Overall screen layout – tester should understand the reasoning behind choices of LOVs, checkboxes, and list items, multi-record vs. single record display, push buttons, and radio groups.
   - Do pull-down menu operations work properly?
   - Are all pull-down menu functions listed correctly?
   - Are character text, size, placing, and format correct?
   - Does each menu function perform as advertised?
   - Does alphanumeric data entered fit within entry restraints?
   - Do graphical modes of data entry work properly?

Example – GUI Testing

- Testing a drop down box on a page to make sure the drop down box populates correctly and that data is selectable.

Test Planning

Test planning must conform to the procedures in **Section 3** of the Handbook.

Test Phases

- **Unit Testing** may require GUI testing as described in **Section 4.2** of the Handbook.
- **Integration Testing** may require GUI testing as described in **Section 4.3** of the Handbook.
- **System Testing** may require GUI testing as described in **Section 4.4** of the Handbook.
- **UAT** may require GUI testing as described in **Section 4.5** of the Handbook.
- **Post Implementation Verification** may require GUI testing as described in **Section 4.6** of the Handbook.

6. Web Based Testing

Web-Based Testing involves testing applications via web browsers. Testing web-based applications is similar to testing any other application in terms of testing the functionality. The differences come from the distributed nature of the web environment. For example, when an error occurs, it is difficult to determine the exact location of the error.

Specific testing areas in web testing include the following:

- **Content Checking**: Inspecting the web application for consistency, accuracy, spelling, and accessibility
- **Browser Syntax Capability**: Even if the web application runs on only one browser (e.g., Microsoft Internet Explorer), the browser syntax must be checked for compatibility since there are many versions of Microsoft Internet Explorer.
- **Functional Testing**: Functional Testing mainly focuses on page level and transaction level testing.
    - Page level testing requires client code to check the field validations.
    - Transaction level testing mainly focuses on whether the information entered by the user at the web page level is updated in the database and proper data is returned to the user.
- **Performance**: Performance testing examines the throughput of the web application as the number of users and/or transactions is increased.
- **Security**: Security testing checks for web application security vulnerabilities.

Example - Web Based Test

An example of Web Based Testing is to validate the consistency of navigation between web pages testing each HTML link.

Test Planning

Test planning must conform to the procedures in **Section 3** of the Handbook.

Test Phases

- **Unit Testing** may require Web Based Testing as described in **Section 4.2** of the Handbook. Validation uses a tool (such as an HTML Validation Service) to validate the HTML syntax and browser compatibility.
- **Integration Testing** may require Web Based Testing as described in **Section 4.3** of the Handbook. Test examples include clicking to check every page and link of the website to ensure content was migrated to the correct page.
- **System Testing** may require Web Based Testing as described in **Section 4.4** of the Handbook.
- **User Acceptance Testing** may require Web Based Testing as described in **Section 4.5** of the Handbook. Test examples include checking the consistency of the look and feel of the web site and checking the ease of navigation.
- **Post Implementation Verification** may require Web Based Testing as described in **Section 4.6** of the Handbook.

7. 508 Compliance Testing

Section 508 of the Rehabilitation Act of 1973 requires Federal agencies to develop, procure, maintain, or use electronic and information technology to ensure that:

- Federal employees with disabilities have access to and use of information and data that is comparable to that available to Federal employees who do not have disabilities, unless an undue burden would be imposed on the agency; and
- Members of the public with disabilities seeking information have access to and use of information that is comparable to that available to individuals without disabilities.

These technology-specific provisions address:

- Software applications and operating systems;
- Web-based information or applications;
- Telecommunications products;
- Video or multi-media products;
- Self contained, closed products such as information kiosks and transaction machines; and
- Desktop and portable computers.

The four-step procedure to carry out 508 compliance testing includes:

1. Developing a test plan based on the Section 508 compliance requirements outlined by the client.
2. Conducting 508 compliance testing by a contractor, or by Federal Student Aid.

3. Conducting 508 compliance testing using the Assisted Technology Partnership Group (ATP) from The Department of Education (AR_Process_Brief_Request_Form_1.0 _ Educatehome.doc).
4. Documenting all feedback from outside the organization according to Section 508 compliance.

The Contractor has full responsibility to conduct 508 compliance testing of the application. The Assistive Technology Partnership (ATP) group will verify compliance after the contractor test team has conducted the 508 compliance testing. The Contractor is also responsible for assisting the ATP Group during their test efforts.

The Contractor must include the following in the Test Plan for 508 Compliance Testing:

- Identify the tool being used to perform 508 testing (e.g., JAWS)
- Coordination and support with ATP to include:
    - Scheduling of tests
    - Minimum two weeks' notice (standard is three weeks)
- On site contractor support

Additionally, the results of the 508 compliance testing are required to be provided to Federal Student Aid separately from other test results, as they will be required during the Production Readiness Review.

Note: *Federal Student Aid will provide Section 508 compliance guidelines at the time of contract award.*

Example - 508 Testing

When software is designed to run on a system that has a keyboard, product functions shall be executable from a keyboard where the function itself or the result of performing a function can be discerned textually.

Test Planning

Test planning must conform to the procedures in **Section 3** of the Handbook.

Test Phases

- **Unit Testing** may require 508 Compliance Testing.
- **Integration Testing** does not require 508 Compliance Testing.
- **System Testing** may require 508 Compliance Testing.
- **UAT** may require 508 Compliance Testing.
- **Post Implementation Verification** does not require 508 Compliance Testing.
- The following Accessibility Review Process Brief provides additional guidance on Section 508 Testing:

ED Accessibility Review Process Brief, a.k.a., Section 508 Testing

This application is being tested to determine how well it complies with the standards established by the Access Board that implements Section 508 of the Rehabilitation Act of 1973 as amended in 1998. For further information on the applicability of Section 508 to the purchase of electronic and information technology, please refer to the ACS Directive, "Procuring Electronic and Information Technology under Section 508 of the Rehabilitation Act": http://wdcrobiis08/doc_img/acs_ocio_3_105.doc.

In conducting this assessment, the Department of Education's Assistive Technology Team will utilize the following protocol:

1. The testing team shall consist of one or more members of the Assistive Technology Team: an expert user of the application/site who is familiar with keyboard shortcuts, familiar with all modules which will be used at Education and how they will be utilized, and someone with sufficient access to the application development team to intelligently transmit the findings of the review to the development team for purposes of remediation. When possible, a member of the actual development team shall be present, as this is the most effective means of ensuring that accessibility problems are fully understood by the vendor. The review will be canceled if someone with the knowledge and skills described above is not present at the allotted time.
2. The set of standards used (Part 36 CFR 1194) will be comprised of the Section 508 Part B Technical Standards that are directly applicable to the type of application in question. For example, all web sites and applications will be evaluated against 1194.22, the "Intranet and Internet-based Applications" standards. All software applications will be evaluated against 1194.21, the "Software Applications and Operating Systems" standards. Those web-based applications that contain non-HTML content will, when appropriate, invoke the software standards when dictated by web standard 1194.22(m).
3. All tests will take place at Education headquarters, 400 Maryland Avenue, Southwest, in Room BC-101 at one of the designated testing machines. Requests to review applications at locations other than our testing area will be considered on an individual basis and only honored as staff resources permit.
4. In consideration of those with chemical sensitivity, the Assistive Technology Team requests that all persons attending this meeting refrain from wearing any scented products including perfumes, cologne, aftershave, hairspray and other strong fragrances. For more information regarding Multiple Chemical Sensitivities, please reference: http://www.access-board.gov/about/policies/fragrance.htm
5. ***ED or vendor personnel requesting an accessibility review should allow at least two weeks' lead time to schedule the review after contacting the team.** To schedule a review, please contact Mark Loeffler, Accessibility Review Coordinator, via email or by telephone (202-377-4938), or via our main voice number (202-260-5055) or shared TTY line (202-401-8510). Requests for emergency reviews requiring a quicker turn-around will be honored as staff resources permit.
6. Tests will last up to but be no longer than two hours in length. For in-depth, complex applications that may require additional time, more than one session may need to be scheduled to complete the review process.

7. ***Some applications may require passwords and user names to be supplied, firewall access to a development network, specialized installations of software or drivers on the test machine, or similar pretest arrangements. It is the responsibility of the ED application sponsor requesting the review to work with engineering and assistive technology staff to ensure that all such preparatory actions have been completed prior to the time of testing. All tests for which such preparations have not been completed will be canceled and rescheduled for a future time.**
8. As stated above, when possible, an individual representing the company responsible for developing the application being tested, with sufficient knowledge of programming to communicate specific standard-related findings back to company development staff, shall be present. This is to ensure that detailed information concerning any non-compliant user interface elements discovered during the testing process can be adequately communicated to development staff, making remediation as efficient and effective as possible.
9. Due to the size and complexity of most client and web-based applications or sites, only a representative sample of the user interface elements will be tested during the formal review process. It is the sole responsibility of ED staff and vendor representatives present at the review to guide the testing team to samplings of all user elements relevant to the applicable standards. To assist in this process, the testing team will verbalize and explain each applicable standard, requesting of the project or company representative that samples of each standard be evaluated during the review process. *** Failure to draw attention to controls or elements that are later found to be non-compliant will not exempt these elements from the requirements imposed by the Section 508 standards.** It is for this reason that we require someone present who is extremely familiar with all aspects of the application interface, as it will be deployed in the ED environment.
10. Various testing tools will be used to evaluate the accessibility of interface elements, and may consist of screen readers, screen magnifiers, and manual or automated code examination. The selection of tools is at the discretion of the testing team, which will choose the tools most appropriate for the given situation.
11. A report briefly outlining the findings of the team will be prepared within three business days of the review. If the application fails 508 compliance testing, it will be the sole responsibility of the ED sponsor to proceed with one of the following options: a. require remediation and retest of the application prior to deployment; b. choose a product which conforms to the Section 508 standards; or c. invoke a Section 508 exception and inform the CCRB of the decision to implement. The vendor, in consultation with the ED project manager, shall make every attempt to remediate inaccessible application elements that are discovered to be non-compliant after the formal testing process. Members of the Assistive Technology Team will be available to assist development staff on a limited basis, providing consultation on the standards and testing methods during the remediation process.
12. All applications that are installed on the Assistive Technology workstation used for testing will reside on the machine for a period not to exceed two weeks. When remediation requires a time period between the initial test and the retest that exceeds this

time period, the ED project manager may be responsible for ensuring that the application is reinstalled on the test machine.

13. All upgrades, updates, and new versions of client-server and web-based applications that make any change to the user interface shall be tested for compliance.
14. The results of all accessibility reviews are considered to constitute ED's best understanding of the application of the Section 508 implementing standards, and are provided as advisory guidance to ED customers and vendors. The decision to deploy a given application which does not conform to the set of standards against which it is tested rests solely with the ED project manager after seeking advice from the Assistive Technology Team in such areas as: remediation, potential liability, Section 508 exceptions which may apply, etc.
15. For a full discussion of the Assistive Technology Team's Accessibility Review Process, please reference: http://educatehome/EDUCATEDocuments/General%20Access/EDUCATE-PRO-000-0414_v1.0_Accessibility_Review.pdf

For any questions about Section 508, please contact Don Barrett, Section 508 Coordinator (202-377-4079) or call the main number for the team 202-260-5055 (TTY: 202-401-8510).

8. Commercial-off-the-Shelf (COTS) Software Testing

COTS software is commercially available software sold with the expectation that the source code will not be available and that documentation will be limited to installation and end user instructions and information. Federal Student Aid integrates COTS software with a number of Federal Student Aid applications. Testers use Black Box Testing techniques since there is no access to the product's source code.

Before integrating COTS software, Federal Student Aid must have a thorough understanding of the compatibility of the COTS product. In most integrated COTS software systems, Functional Testing is limited. When working one-on-one with the contractors or developers of COTS software, developers must follow a thorough testing process to insure that software complies with requirements of the COTS software. Access to thorough documentation and a help desk can mitigate the risk and reduce testing failures that can occur when integrating third party software.

One of the least tested but most critical features of software applications is error/exception handling. Error/exception handling routines are the safety net for any system to handle unexpected circumstances, such as when operating system software or hardware failures occur. As more critical applications are developed using or through integration with commercial-off-the-shelf software, the sensitivity of these applications to operating system failures, and in general to failures from third party software, becomes increasingly critical. By working closely with the developers and having proper documentation and test results, these risks can be limited. Maintaining newer versions of the COTS software reduces the number of known issues in the production environment.

Example - COTS Testing

An example is testing the integration of a COTS product (e.g., WebSphere MQ) that is installed to enable two applications to transfer data from one to the other. These tests would also include evaluating the COTS product's performance when one of the two applications goes down for a period of time and then comes back up, to evaluate how well the data transmitted by one of the applications during that period is transmitted once both applications are operating normally.

Test Planning

Test planning must conform to the procedures in **Section 3** of the Handbook.

Test Phases

- **Unit Testing** is described in **Section 4.2** of the Handbook.
  - Unit Testing is only required when the custom code of the COTS product is modified.
- **Integration Testing** may require COTS Testing as described in **Section 4.3** of the Handbook.
- **System Testing** may require COTS Testing as described in **Section 4.4** of the Handbook.
- **UAT** may require COTS Testing as described in **Section 4.5** of the Handbook.
- **Post Implementation Verification** may require COTS Testing as described in **Section 4.6** of the Handbook.

9. Client Server Architecture Testing

Client Server describes the relationship between two computers in which a program in one computer, the client, makes a request to another computer, the server. Typically, servers are specialized computers dedicated to a single task, e.g., managing disk drives (file servers), printers (print servers), or network traffic (network servers). Clients are PCs or workstations on which users run applications. Clients may rely on servers for resources, such as files, devices, and even processing power.

Example - Client Server Testing

An example of Client server architecture testing is testing concurrent query updates. The following scenarios can be tested with a client installed on two different machines:

- Both clients access the server at the same time – One updates a record and the other tries to access the same record
- Both clients access the server at the same time – One deletes a record and the other tries to access the same record

Test Planning

Test planning must conform to the procedures in **Section 3** of the Handbook.

Test Phases

- **Unit Testing** may require Client Server Testing as described in **Section 4.2** of the Handbook.

- **Integration Testing** may require Client Server Testing as described in **Section 4.3** of the Handbook.
- **System Testing** may require Client Server Testing as described in **Section 4.4** of the Handbook.
- **UAT** may require Client Server Testing as described in **Section 4.5** of the Handbook.
- **Post Implementation Verification** may require Client Server Testing as described in **Section 4.6** of the Handbook.

10. Security Testing

The MTP must state that Security Testing guidance will be included in the Certification & Accreditation plan. When there is no need for security related testing, the MTP must explain the reasoning for exclusion.

Information security is a top priority within Federal Student Aid. Security testing involves using appropriate methods and procedures to validate the implementation of security controls on the application and/or the environment in which the application operates. Test results show the extent to which the controls are correctly implemented, operating as intended, and producing the desired outcome and accurate implementation of the security requirements of the system.

Security Testing includes the following areas:

Table D-3, Vulnerability Scanning

Area	Responsibility
All teams must contact the Virtual Data Center to schedule security scans.	
Upgraded or Downgraded Server(s) and/or Application(s)	If a server(s) or application(s) is upgraded or downgraded, a vulnerability scan must be performed A.S.A.P.
New or Upgraded Server(s) and/or Application(s)	Several tools are run for new or upgraded servers and applications. These tools check for vulnerabilities against the Common Vulnerabilities and Exposures (CVE) database, which is maintained by Mitre for US CERT. The tools also check for misconfigurations based upon the NIST hardening guidelines.
Vulnerability Scripts	Scripts are created based on the features and settings within a given product.
WEB Servers and WEB Applications	Web Inspect is used for WEB servers and WEB applications. As a double blind product, Shadow Security Scanner and/or Nexus may be used.
Databases	AppDetective is used to conduct scans on databases.
Generic Servers and Applications	Shadow Security Scanner and/or Nexus is used on generic servers and applications.

Example - Security Testing

Password Cracking

During these tests, testers identify weak passwords. A strong password contains 10 or more characters that include upper case and lower case letters, numbers and special characters.

Examples of negative Security Tests include the following:

- Trying to access a restricted URL by having a user who is logged in but is not authorized to go to the restricted URL type the restricted URL into the browser's URL window.
- Testing of users' privileges to parts of the applications (no access, read-only access, write access, create, read, update, delete, etc.).

The following links contain details of Security Testing for Federal Student Aid:

- An index of the policies and procedures governing information security controls required for Federal Student Aid is located at: http://thestartingline.ed.gov/cio/products/it_security_portal/index.shtml
- Policies, procedures, templates, checklists, forms, training materials and links to internal department websites related to information security are located at: http://thestartingline.ed.gov/cio/products/it_security_portal/library.shtml

Details on Test Planning and Testing Phases can be found by following the above-mentioned links as well as the National Institute of Standards and Technology (NIST) Special Publication 800 series.

11. Disaster Recovery of Continuity of Services (CoS)

The goal of disaster recovery testing is to ensure that if a system experiences the worst-case scenario of a failure (hardware or software) at the VDC, the system is able to fully recover and resume normal business operations at a Federal Student Aid designated backup facility.

Federal Student Aid has implemented the *Virtual Data Center (VDC) Continuity of Services Plan*, which defines the resources, actions, tasks and data needed to recover systems and the infrastructure that supports those systems.

The *Virtual Data Center (VDC) Continuity of Services Plan* contains the standards for disaster recovery testing.

Contingency test plans must be created and are a separate document from the disaster recovery plan.

12. Review and Testing of Documentation

Software documentation and the testing of software documentation is an important step in understanding and accepting software. Many federal organizations have special teams of "document reviewers." Contractors must understand that documentation is just as important as the functional aspects of their software.

Testing of documentation is a necessity, but this process is not complete until all of the necessary documents have been tested and reviewed. Having thorough documentation from installation guides to help files, test plans to test summaries, as well as thorough software documentation, reduces confusion and may reduce software-testing time.

Some important documentation to consider includes:

- **Requirements, Business Use Cases, System Use Cases, User Guides, and Detailed Design Documents**: Define how functionality will be implemented, and how to use the application.
- **Test Strategy**: Outlines the high-level process for testing the application.
- **Results Summaries**: Identifies the results of each round of testing.
- **Known Issues Document**: Used primarily for Technical Support and identifies issues Development is aware of but has chosen not to correct, potential problem areas for users, and workarounds.
- **Open, Deferred or Pending Defect Summary**: Summarizes the defect ID, the problem area, and title of defects remaining in the defect management system with a status of Open, Deferred or Pending.
- **Fixed Defects**: Lists defects waiting for verification by Quality Control.
- **Installation and Help Instructions**: Aids system administrators in installation, integration, and trouble-shooting installation problems and identifies how to access help files and where links are displayed for testers and users.

These documents all facilitate the Quality Control process and explain, at a high level, the key documents that must be received, installed, or utilized during any project cycle, integration process, or installation process, in order to assess the readiness of a product for release. Although the level of detail in each document may vary by contractor and test organization, each provides evidence of appropriate testing, test management, and project status.

Example – Testing Documentation

The following scenarios can be applied to testing user guides and manuals:

- Is the information provided technically appropriate for readers of this manual?
- Are screen shots and other content up to date?

Test Planning

Test planning must conform to the procedures in **Section 3** of the Handbook.

Test Phases

- **Unit Testing** may require Review and Testing of Documentation as described in **Section 4.2** of the Handbook.
- **Integration Testing** may require Review and Testing of Documentation as described in **Section 4.3** of the Handbook.
- **System Testing** may require Review and Testing of Documentation as described in **Section 4.4** of the Handbook.

- **UAT** may require Review and Testing of Documentation as described in **Section 4.5** of the Handbook.
- **Post Implementation Verification** may require Review and Testing of Documentation as described in **Section 4.6** of the Handbook.

13. Test Automation

While manual testing is routinely used for some tests, automated testing tools are also a popular way to assess and apply faster Regression and Load Testing to validate a website's performance and stability. The contract or statement of work will identify the tools, or types of tools, to be used for testing, including the name of the publisher and the version number to be used.

Test Suites that contain automated test scripts should be separate from test suites that will be executed manually. The manual tests may be outside the range that the automated tests can execute. In addition, separating the two test types is important to maintain discrete metrics. For example, the number of pass and fail attempts using automated test scripts is generally much larger and should be kept separate from the number of pass and fail attempts that occurred during manual testing.

Test Automation includes the use of software testing tools to control the execution of tests, to compare actual outcomes to predicted outcomes, to create test preconditions, to create test data, and to perform other test control and test reporting functions. Creating automated tests frequently involves automating an existing manual process.

Advantages of Test Automation

There are several advantages to test automation:

- Low Time to market and reduced testing time because scripts only need to be created or modified when changes occur in the related requirements
- Improved Testing productivity because the Testing Team doesn't need to repeat the same test suites manually
- Improved Product Quality

Types of Tools Used

Although Federal Student Aid does not require the use of a specific tool, Mercury LoadRunner has been used for performance testing at Federal Student Aid. The Project Manager and Test Manager will approve the tool proposed by the contractor. Some other types of tools include:

- **Automated Regression Testing Tool**: Used primarily as a capture-playback testing tool that records test scenarios, plays them back, and generates reports comparing test results with reference results.
    - These tools can also be used to extract data for input to test suites (i.e., Rational Functional Tester and Rational Robot).
- **Defect Management and Related Tools**: Record defects identified during testing.

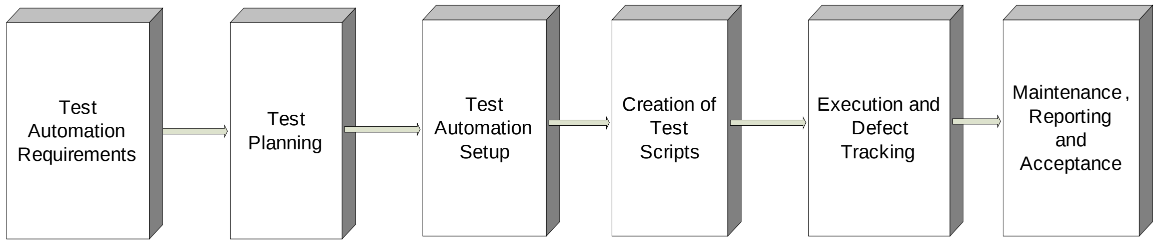
  - Maintain related information including the state of the defects until the issues have been resolved (i.e., Rational Clear Quest).
- **Manual Tools**: A simple, yet effective, checklist that enables testers to ensure that test activities have been performed correctly.
  - Manual tools are used primarily during reviews and inspections.
- **Traceability Tools**: Used to associate baselined artifacts produced in various phases of the software development lifecycle so that the effect of changes can be visually represented.
  - An example is Rational RequisitePro.
- **Code Coverage Tools:** Indicate the amount of code executed during testing.
  - Some of these tools can also identify non-entrant code (i.e., Rational PurifyPlus and Clover).
- **Test Suite Management Tools**: Assist in test planning, designing, implementing, executing, evaluating, and managing test activities or artifacts (i.e., Rational Test Manager).

The Rational Suite is the preferred Federal Student Aid tool for testing. If Rational was used for requirements management, the same should be used for testing.

Test Automation Life Cycle

The Test Automation Life Cycle Process demonstrates the life cycle:

Figure D-1, Test Automation Life Cycle Process


Test Automation Requirements
Test Planning
Test Automation Setup
Creation of Test Scripts
Execution and Defect Tracking
Maintenance, Reporting and Acceptance

Test Automation Requirements

Before testing can be automated, the following conditions must exist:

- Requirement/Functional Specification documents must exist and be approved
- Design Specification documents (use cases)
- Test Traceability Matrix for identifying Test Coverage
- Functional/Non-Functional and test data requirements

Planning

If test automation is to be performed, the following test planning issues must be addressed:

- Automated Software Testing: Scope must be defined to identify the area and percentage of automation
    - Automation should be attempted first only on a subset of a system
- Identification, evaluation and procurement of the automated testing tool
- Creation of an automated testing Framework

Test Automation Environment Setup

To create an automated testing environment, the tools required for automation testing must be installed on the appropriate servers/machines.

Design

Once the environment has been created, testing with an automated testing tool requires:

- Preparation of test scripts
- Unit testing of the test scripts
- Integration Testing of the test scripts

Execution and Defect Management

Following the design of tests, execution of the tests and defect management requires:

- Execution of the automated test suite
- Analysis of the Test Results
- Defects reporting and tracking

Maintenance, Reporting and Acceptance

Following testing, defect management, and defect resolution, results should be reported as follows:

- Automated Software Test Results and summary reports
- Test Reports consisting of metrics
- Getting Acceptance

Example - Automation Test

An example of Test Automation is GUI Regression testing using a testing tool. These tools can create customizable, proprietary scripts by recording User Interface interactions in a test case during an interface testing session. The scripts can be played back to repeat these User Interface interactions, thereby eliminating the need for the tester to manually repeat User Interface interactions.

14. Application Programming Interface (API) Testing

An Application Programming Interface (API) is specialized software providing functionality that is executable by other software. APIs greatly reduce the need for individual applications to perform fundamentally generic functions like some printing and security functions, and database access.

API Testing is performed using test frameworks for Unit Testing, Integration Testing, and Regression Testing performed prior to System Testing. During UAT, the business users test the application functionality that incorporates the API calls.

Example – Application Programming Interface

An example of API testing is Reusable Common Services (RCS) testing within Federal Student Aid. RCS components are reusable components that application groups can use to lessen programming effort, enforce standard modes of business, and implement best practices.

15. White Box Testing

White Box Testing strategy helps to ensure the internal logic and structure of the software's code. White Box Testing is also known as Glass, Structural, Open Box, or Clear Box testing. Tests written using a White Box Testing strategy include coverage of the statements, decisions, conditions, and internal logic of the code.

White Box Testing Types

Statement Coverage

Statement coverage helps ensure that all the statements in the code execute correctly without any unintended side effect. While statement coverage tests the execution of each statement, it does not control structures such as Boolean expressions (for example, determining if an "if" or "while" statement provided the correct result).

Decision Coverage

Decision Coverage is used where Boolean expressions are tested (for example in testing "if" or "while" statements). This method should be used to test both true and false conditions.

Condition Coverage

Condition coverage is used when code contains Boolean expressions that include branches. For example:

Example 1 simplified

If (number of apples > 5 *OR* (number of oranges > 2 AND number of pears > 5))
then [do this] else [do that].

Decision coverage could ignore execution of Function1 whereas the condition coverage reports the true or false outcome of each Boolean sub-expression, measuring each one of them independently.

Example 1 as coded

```
If (expression1 || (expression2 && Function1() ))
        Statement 1
Else
        Statement2
```

Example – White Box Testing

An example of White Box Testing is stepping through the code, line by line, to determine whether the code module responsible for login requests properly authenticates valid requests and rejects invalid requests.

16. Black Box Testing

Black Box Testing tests the functionality of a program or application against its specification without knowledge of or direct review of the underlying code or internal logic.

Black Box Testing Types

Equivalence Partitioning

Input data and output results often fall into different classes where all members of the class are related. Each member of this class is called an Equivalence Partition. The process of determining these input classes is defined as Equivalence Partitioning.

Boundary Value Analysis

Boundary value analysis is a technique that consists of developing test suites and data that focus on the input and output boundaries of a given function. Boundary values include maximum, minimum, just inside/outside boundaries, typical values, and error values. The belief is that if a system works correctly for these special values, the system will work correctly for all other values inside and outside the boundaries.

Error Guessing

Error guessing involves developing an itemized list of the errors expected to occur in a particular area of the system, then designing a set of test cases to check for these expected errors. Error guessing is more of a testing “art” than “science” but can be very effective when implemented by a tester familiar with the history of the system.

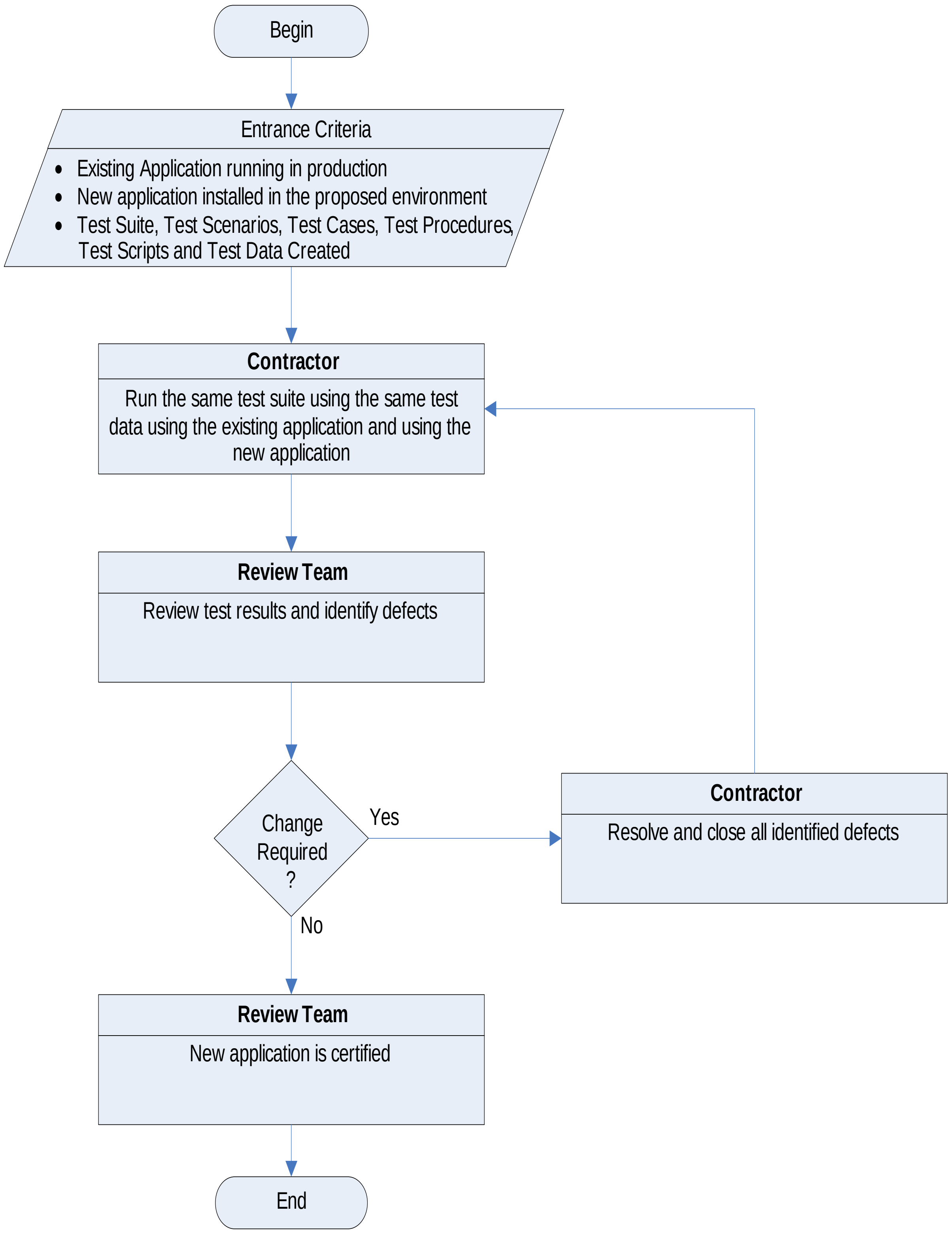
Example – Black Box Testing

Testing input values for a student’s Social Security Number on the application’s registration page to determine whether the characters defined as acceptable are accepted, the characters defined as unacceptable are not accepted, and the appropriate error message is returned.

17. Parallel Testing

Parallel Testing involves operating a new application and an existing application in parallel for a specified period. Each system receives the same data input and performs the same or equivalent workflow with both applications producing the same results. **Figure D-2** on the following page illustrates the parallel testing process:

Figure D-2, Parallel Testing


Begin
Entrance Criteria
• Existing Application running in production
• New application installed in the proposed environment
• Test Suite, Test Scenarios, Test Cases, Test Procedures, Test Scripts and Test Data Created
Contractor
Run the same test suite using the same test data using the existing application and using the new application
Review Team
Review test results and identify defects
Change Required ?
Yes
Contractor
Resolve and close all identified defects
No
Review Team
New application is certified
End

18. Regression Testing

Testers use Regression Testing to evaluate whether previously tested functions or components still function correctly. Regression Testing is important during application development and maintenance of an exiting application. This testing must ensure that code changes have not inadvertently affected functionality of the application.

The most effective Regression Testing involves developing test suites that test common functionality and can be executed every time a new version of the program is built; it is also important that existing functionality is thoroughly tested when new fixes are applied to the software. A thorough Regression Test suite can validate that a change to the program has not inadvertently changed any other application functionality.

Example - Regression Testing

1. When a SSN entry defect occurred in production, the application was updated to resolve the SSN entry defect. The entire SSN entry page was retested to ensure that no other functionality on that page was affected in the process.
2. A brand new application was created to create a new student database. The System Test Team found a defect while entering the student's last name. All other functionality for the new application worked according to the design. Developers corrected the student last name defect. The System Tester tested the application to ensure that the student last name defect was corrected and that all other functionality for the application still worked according to the design.

19. Agile Testing

The testing practices that follow the Agile Manifesto, **Figure D-3, Manifesto for Agile Software Development**, treat development as the customer of testing. In this light, the context-driven manifesto provides a set of principles for Agile testing. Testing practices for projects using Agile methodologies redefine the role of the tester on an Agile project.

Figure D-3, Manifesto for Agile Software Development

Section 7

Manifesto for Agile Software Development

(http://www.agilemanifesto.org)

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on
the right, we value the items on the left more.

Agile provides a continuous stream of value to customers, or Application Owners. The two most important aspects of Agile are:

- Increase the rate of feedback
- Reduce the waste

Agile can be described in terms of what it is not. Agile is not about:

- Compressing the schedule, nor
- Tossing out the documentation, and certainly not
- Coding up to the last minute.

It is all about maximum value for minimal cost.

If a decision has been made to use one of the Agile development methodologies mentioned below, it is important to have the Application Contractor help the Application Owner to understand the approach and all of the associated processes and concepts. Some of the terms used are specific to Agile methodologies and require a level of understanding that may require more support from the contractor.

Overview

Since Development and Testing are tightly linked in Agile Methodologies, there is no separate approach to Development versus Testing. They are one and the same and require continuous participation from the Application Owner. There are a number of development methodologies that are considered to fall under the "Agile" umbrella?

- Extreme Programming (XP)

- Crystal
- Adaptive Software Development (ASD)
- Scrum
- Feature Driven Development (FDD)
- Dynamic Systems Development Method (DSDM)
- XBreed

XP projects, for example, are finding new ways to build in testability and support automated tests. “Agile” approaches use methods requiring close interaction and cooperation between programmers and Application Owners (customers/end-users) to iteratively develop requirements. In the XP “test first” approach, developers create automated unit testing code before the application code, and these automated unit tests essentially embody the requirements.

Impact of Agile on Test Planning

There are three key principles of Agile testing:

1. The system always runs
2. No code is written without a failing test
3. Zero post-iteration bugs

There are three key strategies to help implement these principles:

1. **Communicate** - Agile development depends on disciplined communication around requirements and testing.
2. **Commit** - If the team doesn’t commit to delivering flawless software every iteration… they won’t deliver flawless software every iteration!
3. **Automate** - If testing is not significantly automated, we cannot easily check whether the system is running. It is important to note that with any of the Agile methodologies, automation is typically a requirement of the selected approach given that the main focus of these methodologies is a quick, iterative process. There are many aspects of the approaches that need to be tracked and monitored, and a manual process will introduce more risk.

Role of Testing

The following is a list of roles for testing in Agile environments.

- Testing is the headlight of the project, indicating, “Where are you now?” and “Where are you headed?”
- Testing provides information to the team, allowing the team to make informed decisions.
- A “bug” is anything that could complicate user interactions; however, the testers do not make the final call to revise the issue.
- Testing does not assure quality; the team does (or does not).

- Testing is not a game of "gotcha" but is an approach for finding ways to set goals, rather than focus on mistakes.

In general, regardless of the Agile Development Approach selected (i.e., Extreme Programming (XP), Crystal, Dynamic Systems Development Method (DSDM)), Agile testing happens at two levels, and only two levels. There are no traditional phases of development or testing in an Agile Project.

Developer Tests define whether the system does what the developers expect ("building the code right"). Developer Tests Drive Design (TDD):

- Testing of the design occurs mostly as Unit Tests but can include integration tests
- Written cooperatively by developers and testers
- Define how developers intend the system to behave
- Enable developers to verify that, after a change, the system still works the same way
- Guide programming and design
- Can be run automatically at any time by developers

Developer Testing in an Agile environment is challenging because:

- It is fundamentally a shift in thinking; Developer TDD is a change in mindset for developers
- Writing tests takes time – initial dip in developer "productivity"
- Many developers are reluctant to write tests
- Tests require disciplined maintenance
- Tempting to fall back into old way of working
- Whole team must buy in
- Requires tight build/integration

Acceptance Tests tell us whether the system does what the customer expects ("building the right code"). In Agile development methodologies, testing practices are important. All features must have **automated acceptance (functional) tests.** All tests (acceptance and unit) must run with a binary pass/fail result, so that no human inspection of individual test results is required. The acceptance tests are written with collaboration of the Application Owner and they define a testable statement of what acceptance means.

In the event that the Agile test methodology is used, Federal Student Aid may perform the testing itself or have IV&V or a test contractor perform the test. The Federal Student Aid project manager makes the decision if User Acceptance Testing is required.

The Acceptance Tests in an Agile development environment:

- Define how the Application Owner wants the system to behave, and are specified by the Application Owner, and eventually enable the Application Owner to verify that the system works as required;

- Define business requirements and are also called customer tests or functional tests;
- Serve as executable requirements documents, however they specify business requirements, not application design;
- Can be understood by non-technical people;
- Can include Integration Tests, Smoke Tests, Performance Tests, Load Tests;
- Enable Developers to know they've satisfied requirements;
- Can be run automatically at any time by anyone; and
- Should be independent of implementation.

Creation of Acceptance Tests must be collaborative. Acceptance criteria are specified by the Application Owner at the start of an iteration. Whenever possible, the Tester writes executable acceptance tests. If needed, testers work with Application Owner to formalize criteria into scenarios and acceptance tests. Testers make those scenarios executable using tools. Developers help testers connect tests to the actual system. Tests are written just before the developers implement a Feature (a functional piece of the code).

Agile Acceptance Testing is a fundamental shift in thinking and requires a change in how projects are run. It can be difficult to automate acceptance tests, and many application owners do not know how to write acceptance tests. This is where the collaborative nature of the approach is very evident. More information and participation is required from Application Owners than in any of the more traditional development approaches. In an Agile environment, testers need to be involved earlier and the tests require disciplined maintenance as they must be able to be run automatically by anyone. It is tempting to fall back into old ways of working. The entire Agile development approach requires a tight build/integration process and rigorous management of the iteration.

Roles and Responsibilities

The primary Roles and Responsibilities of an Agile Development Project lie with the Application Owner and the Development Team. Since the Development Team is also responsible for the testing, the Test Team as we know it in other development methodologies does not exist separately. A tester might facilitate the Acceptance Testing sessions with the Application Owner, but since there are no discrete phases for development or for testing, there is no specific role for the tester outside of the development of the product. The following table shows the roles for the Application Owner and the Development Team (which included the "testers").

Table D-4, Application Owner and Development Team Responsibilities

Application Owner Responsibilities	Development Team Responsibilities
Review the release plan to make sure the vision and goals are still appropriate.	Review the top priority items in the backlog and prepare any questions.
Review the items in the backlog and reprioritize if necessary.	
This includes stories (scenarios) that (a) were already there (originally defined in previous release planning sessions); (b) have been added	Consider technical issues, constraints, and dependencies, and be prepared to share these concerns.

Application Owner Responsibilities	Development Team Responsibilities
since the last release planning session; (c) failed acceptance in a prior iteration; (d) are generated from defects or bugs.	
Understand how the reprioritization may affect other teams who are dependent on a deliverable committed to during release planning. Coordinate with other product managers as needed to resolve dependency issues.	Think about the work involved in delivering the functionality in the stories, in order to be prepared to make estimates in the meeting.
Understand the customer needs and the business value that each story is to deliver.	Understand what your iteration velocity should be for the upcoming iteration, based on team discussions at the last review.
Be prepared to further elaborate on the details of each story.	

Defect Management

In contrast to more traditional development methodologies, the tests are "written" and executed in a different sequence in Agile development projects.

- Traditional: After development is complete, at the end of the project
- Transitional: After development is complete, at the end of each iteration
- Agile: No code is written except in response to a failing test

Likewise, defects are managed in different ways.

- Traditional: Identified during test and logged in a defect tracking tool
- Transitional: Identified during development and logged to be fixed later
- Agile: Zero post-iteration bugs, new features are never accepted with defects

Metrics

Progress, effort and cost can be measured in an Agile development project similar to measurements in more traditional methodologies. However, there are fewer static points at which to measure in an Agile development effort. For example, the iteration progresses with a "build a little, test a little" pattern that does not allow for explicit intervals of measurement of defects. As a "defect" is found as a result of a "test," it is corrected by the developer/tester.

Documentation

For Agile software projects, it should be kept in mind that one of the Agile values is "Working software over comprehensive documentation," which does not mean "no" documentation. Agile projects tend to stress the short-term view of project needs, and documentation often becomes more important in a project's long-term context. Tracking all the iterations becomes an important part of Project Management.

With the goal of getting to code fast, XP discourages writing *unnecessary* requirements, design, or management documents. The use of small paper index cards is preferred for jotting brief descriptions, as are verbal communication and elaboration. Note that the practice of "avoiding documentation" is compensated for by the presence of an onsite customer. XP is not anti-documentation, but notes it has a cost, perhaps better spent on programming.

To create documentation for maintenance purposes in an Agile development environment, it is important to define what is needed rather than speculate. If anyone has maintained a prior version of the application, it might be helpful to see what has been useful in the past. Some tips for documentation in this type of development environment include:

- Put documentation on a project website.
- Within many systems, there are a few key tricky or subtle elements or themes that you would not want to lose. Find those, highlight them, and write a short "technical memo" page for each on the project website.
- It is usually useful to document different architectural views.
- Sometimes documentation can be a digital capture of a whiteboard rendering. The development team can be broken up into sub teams, each diagramming different views: the logical view of the architecture, deployment view, security view, etc. Each sub team would create the diagram with some related notes emphasizing the key noteworthy elements in that view.

20. Risk Based Testing

Overall, testing is the means used in software development to reduce risks associated with an application or system. The goal of testing is to identify many of the problems before they get to production, thereby reducing the system's risk. Unfortunately, testing alone cannot find all of the bugs, and with the rapid pace of application development, testing has become a challenging proposition.

Trying to meet even tighter deadlines while still delivering applications that meet requirements is the greatest challenge for testers. Formulating answers to questions like "What should we test?" and "How long do we test?" requires different strategies in fast-paced environments.

- Does the application meet our quality expectations?
- Is the application ready for users?
- What can we expect when 2,000 people hit the site?
- What are we risking if we release now?

Risk is the possibility of suffering harm or loss. In software testing, we think of risk on three dimensions:

- A way the program could fail
- How likely it is that the program could fail in that way
- What the consequences of that failure could be

Risk analysis assesses damage during use and usage frequency, and determines probability of failure by looking at defect introduction. Testing is always a sample. You can never test everything, and you can always find more to test. Thus, you will always need to make decisions about what to test and what not to test. The general goal is to find the worst defects first, the ones that NEED TO BE FIXED BEFORE RELEASE, and to find as many such defects as possible.

This means the defects must be important. The problem with most systematic test methods, like white box testing, or black box methods like equivalence partitioning, boundary value analysis or cause-effect graphing, is that they generate too many test suites; some of which are less important. A way to lessen the test load is to find the most important functional areas and application properties. Finding as many defects as possible becomes more likely when more testing occurs in known problematic or error prone areas of the application. This means you need to know where to expect more defects.

At the Test Suite execution level, the Test Suites are prioritized according to their importance.

- High priority Test Suites will be executed first
- Next medium priority Test Suites will be executed
- Finally, low priority Test Suites will be executed

Typically, high priority will be given to functional requirements test suites; medium priority will be given to non-functional requirements test suites; and low priority will be given to user interface requirements test suites. At the higher project planning level, Risk Based Testing means how adaptively the testing process will be changed to overcome the risk.

In general, the Federal Student Aid process for categorizing risk for purposes of scheduling testing includes the business analysts, lead testers, and application owners as participants. The list of high priority functions of the application should not take much time to create, given the participants' knowledge of the application, and the prioritization should be based on a focus of the critical areas below:

- Business functions that are performed most often
- Areas of the application that most likely contain show-stopping defects. This may be based on complex requirements, requirements that required more clarification than others or past history (areas that have caused problems in the past)
- Functions that are critical to the business
- Functions that are federally mandated
- Defects released in applications that are exposed to outside entities, can have an adverse effect on Federal Student Aid depending on whether the errors are "limited, serious, or severe."
- Critical functions of interest to senior management and stakeholders.

Some functions may be listed more than once. When this happens, those functions should be listed as a high priority for testing, and risk should be documented along with each function.

Once your list is complete, determine if there are any functions that can be tested at a later time. The total list must be prioritized. In the end, the group should agree to the test suites that will be created for the prioritized functions.

Appendix E - Templates

Appendix E: Templates

This section includes a combination of 22 template documents, instructions and forms:

- Document templates required to support Federal Student Aid's testing requirements;
- Template instruction sets for completing the attached documents;
- A common set of formatting instructions for the templates; and
- Template forms.

The template instruction sets are:

Template 1, Master Test Plan – provides a central artifact to govern the planning and control of the test effort. It defines the general approach that will be employed to test the software and to evaluate the results of that testing, and is the top-level plan that will be used by managers to govern and direct the detailed testing work.

Template 2, Test Readiness Review Report – a report for the multi-disciplined technical review to ensure that the subsystem or system under review is ready to proceed into formal testing. This review assesses test objectives, test methods, and procedures, scope of tests, traceability of planned tests to program requirements and user needs and safety, and confirms that required test resources have been properly identified and coordinated to support planned tests.

Template 3, Phase Level Test Plan(s) – document that describes scope, approach, resources, and scheduled of intended test activities for a single test phase.

Template 4, Test Suites – document that outlines the set of several test scenarios, test suites, test procedures and test scripts for a component or system under test.

Template 5, User Acceptance Test Plan – plan that outlines the formal testing with respect to the Application Owner's needs, requirements, and processes conducted to determine whether a system satisfies the acceptance criteria and to enable the user, customers, or other authorized entity to determine whether to accept the system.

Template 6, User Acceptance Test Support Plan – the support plan describes the items necessary to conduct Acceptance Testing; and outlines contractor responsibilities and provides guidance for how the contractor supports Federal Student Aid.

Template 7, Post Implementation Verification Test Plan – test plan that documents the final phase of testing that occurs after the application is deployed in production.

Template 8, Test Summary Report – document summarizing testing activities and results; contains an evaluation of the corresponding test items against exit criteria.

Template 9, Performance Test Plan – documents how to create and gain approval of the mechanisms used to do a variety of types of tests to gather these kinds of statistics:

- Network and/or Web server throughput limits
- Individual server resource utilization under various loads
- Search speeds, query optimization, table/row locking , and speed versus capacity measurements for databases
- Load balancing overhead / capacity / effectiveness
- Speed and resource cost of security measures

Template 10, Performance Testing Test Report – the output of Performance Testing is a collection and presentation of data in ways that show whether the system is missing or meeting goals, in what areas, and by how much, without requiring the viewer of this data to have any special technical knowledge of the system under testing.

Template 11, Hardware Test Plan – the Hardware Test Plan starts with a brief synopsis of the background of the project under test. It also includes topics such as: the schedule information, hardware details, environment location, test facilities that support the test effort, Configuration Management Process, roles and responsibilities, assumptions and constraints, and Risks and Lessons Learned.

The template forms are:

Template 12, Accessibility Review Request Form

Template 13, Performance Test Daily Reports

Template 14, Test Schedule and Effort Variances

Template 15, Test Defect Metrics

Template 16, Defect Report

Template 17, Defect Severity and Current State

Template 18, Test Execution Report

Template 19, Code Coverage Report

Template 20, Tester Team Turnover

Template 21, High Level Test Summary

Template 22, Incident Report By Testing Phase

The following links contain details of Security Testing for Federal Student Aid:

- An index of the policies and procedures governing information security controls required for Federal Student Aid is located at: http://thestartingline.ed.gov/cio/products/it_security_portal/index.shtml
- Policies, procedures, templates, checklist, forms, training materials, and links to internal department websites related to information security are located at: http://thestartingline.ed.gov/cio/products/it_security_portal/library.shtml

Template Formatting Instructions

Component	Description
Overall Guidance	Text enclosed in square brackets and displayed in blue italics (style=InfoBlue) is included to provide guidance to the author and should be deleted before publishing the document. A paragraph entered following this style will automatically be set to normal (style=Body Text).
Template Guidance	*Light-blue text is explanatory and should be deleted from the final document.* Before you start filling out the content of the new document, go to File/Properties/Custom. Enter the Document Name, Version Date (in DD/MM/YYYY format), and Version Number (in x.x format) in the custom properties that bear these respective names. In the document, press CTRL+A, then F9, and click OK. Go into every footer for each section, press CTRL+A, then F9, and click OK. If a section marked "Optional" is deemed unnecessary or inapplicable to the document, the section should be retained and the content for the section should be "Not applicable".
Formatting	**Body Text** The text in the document is the body text style, TNR (Times New Roman) 12, color-black, left-justified. **Numbered Lists (for Sections)** Numbering for lists should follow the [lower-case letter].[number].[number] pattern. All levels in the list will be flush-left with the main text in the section. Example: 1. List item a. List item i. List sub-item **Bulleted Lists (Do not change the font once it is set)** • Bulleted Lists: Level 1 bullets should be Symbol 9 (Format, Bullets & Numbering, Customize, Font-Symbol, Size-9) for level 1 of indentation – black bullet symbol (two spaces in between bullet and first letter of sentence—see ruler at top of document.) Indent first bullet three spaces from left margin.) – ˜Level 2 bullets should be dash symbol for level 2 of indentation. (Font-Symbol, Size-9.) (Two spaces in between bullet and first letter of sentence—see ruler at top of document.) Indent second bullet three spaces from start of Level 1 bullet. ◦ Level 3 should be open bullet symbol for level

Component	Description
	3 of indentation. (Font-Symbol, Size-9) (Two spaces in between bullet and first letter of sentence—see ruler at top of document.) Indent level 3 bullets three spaces from start of Level 2 bullets. **Page setup:** Margins: 1 inch Top, Bottom, Left, and Right; "From Edge"= .5 header, .5 footer. **Section Style Data** Sections should start on their own page. **Heading 1** Font: Arial 16 pt., bold, black Paragraph spacing: "12 pt spaces Before, 18 pt After" Keep with next, Level 1, Outline numbered, tabs 0.5 **Heading 2** Font: Arial 14 pt., bold, black Paragraph spacing: "12 pt spaces Before, 12 pt After" Keep with next, Level 2, Outline numbered, tabs 0.5 **Heading 3** Font Arial 13 pt, unbold Paragraph spacing: "6 pt spaces Before, 6 spaces After" Keep with next, Level 3, outline numbered, tabs 0.5, Indent: 0.5 **Numbered Lists (for Sections)** All major section headings (Heading 1) should be in numerical order (e.g., Section 1, Section 2, Section 3, etc.) Secondary headings (Heading 2) should be in numerical order (e.g., 1.1, 1.2, 2.1, 2.2, 3.1, 3.2, etc.) Sub-secondary headings (Heading 3) should be in numerical order (e.g., 1.1.1, 1.1.2, 2.1.2, 2.1.3, 3.1.1, 3.1.2, etc.)
Tables	You may utilize a few styles for table text and table headings. Caption: Table Caption Heading style: Arial Bold, TNR 12, paragraph spacing "18 pt Before, 3 pt After"

Component	Description
	Table Heading: Arial 11, Bold, paragraph spacing "18 pt Before, 3 pt After" Table Text: Can vary from TNR 9-11 (because documents can vary in size and length, the author may choose which is suitable for his/her document) (The author may choose from Table Text 9, Table Text 10, and Table Text 11 styles.) Bullets within tables: Font of bullet is 6 pt; bullet position: indent at .1", text position: indent at .15" *Example* ***Intended Audience and Document Uses***

Intended Audience and Document Uses

Users	***Relevant Sections***	***Uses***
table text is TNR 11	*table text is TNR 10*	• *table text is TNR 9*
table text is TNR 11	*table text is TNR 10*	• *table text is TNR 9*

Component	Description
Headers	The header contains the title of the document on the left, and the section on the right. **<u>How to add section titles in the header:</u>** 1. Create a section break in the page ahead of the page of the header you want to change. 2. Click View, Headers and Footers. 3. Put your cursor in the header. Ensure that "Same as Previous" is turned off. 4. Click Insert, Cross-Reference. a. In the drop-down box, ensure it states "Heading" and that the "Insert Reference To" states "Heading Text"; then, select the heading in the dialog box that reflects the heading you want to insert. b. Click Insert.
Footers	The footers contain the version number of the document on the left, page numbers in the middle, and the version date on the right. (Regarding version numbers, keep the numbering system to a 1.1, 1.2, etc., and 2.1, 2.1, etc. (For "in-house," you may use alternate numbering systems like "1.1.1" and so forth.)

Component	Description
Appendices	A cover page is required before each Appendix. (Heading 7, TNR 12, centered, paragraph spacing 156 pt “before.”) **<u>Appendix A</u>** Is reserved for acronyms and abbreviations only. List all the acronyms used in the document. Describe all associated terms for this project / document / methodology, etc. Footers: page numbering should read A-1, A-2, etc. *Example*

Acronym	*Definition*
ASG	*Architecture Support Group*

	<u>Appendix B</u> Is reserved for the glossary only. List all of the glossary terms used in the document. Footers: page numbering should read B-1, B-2, etc. *Example*

Term	*Definition*
Common Origination and Disbursement (COD)	*Initiative designed to put in place more common processes across financial aid programs. FSA’s participating schools use a common process, platform, and record to originate and disburse Title IV federal aid funds.*

Additional Appendices	A cover page is required before each Appendix. (Heading 7, TNR 12, centered, paragraph spacing 156 pt “before.”). Additional Appendices should be in sequence to the previous (e.g., Appendix A, B, C, D, etc.). Any appendices created after Appendix A and B are document-specific. Footers: restart page numbering (e.g., if you create an Appendix C, the page numbers in the footers should read “C-1, C-2,” etc.)

Template 1, Master Test Plan

Instructions

Component	Description
All templates must follow the formatting guidelines set forth in the *Template Formatting Instructions* contained in the beginning of Appendix E.	
Cover Sheet	A sample of the cover sheet is provided in the template.
Document Version Control	Provide information concerning the version of the document, the date of the change, and a description of the change.
Table of Contents	A sample of the table of contents is provided in the template.
Project Name	Provide information about the project, including the name.
Executive Summary	• Optional • A one-page high-level introduction describing, in brief: – Document purpose – Document users – Document uses (what, when, how) – Document owners – Document main point of contact for questions or feedback]
Section 1. Introduction	
1.1 Purpose	Describe the purpose of the Test Plan • For example: Provides a central artifact to govern the planning and control of the test effort. It defines the general approach that will be employed to test the software and to evaluate the results of that testing, and is the top-level plan that will be used by managers to govern and direct the detailed testing work. • Include the complete lifecycle, the specific phase, and the project name. • Identify the items that should be targeted by the test. • Identify the levels of testing to be performed (e.g., Unit, Integration, System, and User Acceptance level testing). • Identify the motivation for and ideas behind the test areas to be covered. • Outline the testing approach that will be used. • Identify how defects will be categorized and managed. • Identify the required resources and provide an estimate of the test efforts. • List the deliverable elements of the test project.
1.2 Scope	Defines the types of testing, such as Functionality, Usability, Reliability, Performance, and Supportability, and if necessary the levels of testing, (e.g., Unit, Integration, System [Intersystem], and User

Component	Description
	Acceptance) that will be addressed by this Test plan. It is also important to provide a general indication of significant elements that will be excluded from scope, especially where the intended audience might otherwise reasonably assume the inclusion of those elements. Note: Be careful to avoid repeating detail here that you will define in sections 3, Target Test Items, and 4, Overview of Planned Tests.
1.3 Intended Audience	Provide a brief description of the audience for whom you are writing the Test plan. This helps readers of your document identify whether it is a document intended for their use, and helps prevent the document from being used inappropriately. Note: The document style and content usually alters in relation to the intended audience. This section should only be about three to five paragraphs in length.
1.4 Document Organization	This section briefly describes what is included in the sections and appendices. Section 1 - Introduction: is always the introduction, and a brief description should be provided. Section 2 - [Section Name]: includes the name of the section as well as a brief description. This format must be followed for all of the sections that are included in the document. Appendix A – Acronyms and Abbreviations: this appendix letter is reserved for the acronyms and abbreviations and is a standard titled section. Appendix B – Glossary: this appendix letter is reserved for the glossary and is a standard titled section. Appendix C - [Appendix Name]: includes the name of the appendix as well as a brief description. This format must be followed for all of the appendices that are included in the document.
1.5 References and Related Documents	This subsection should provide a complete list of all documents referenced elsewhere in the Test Plan. Each document should be identified by title, report number (if applicable), date, and publishing organization. Specify the sources from which the references can be obtained. This information may be provided by reference to an appendix or to another document.
Section 2. Governing Evaluation Mission	Provide an overview of the mission(s) that will govern the detailed testing within the iterations.
2.1 Project Context and Background	Provide a brief description of the background surrounding the project with specific reference or focus on important implications for the test effort. Include information such as the key problem being solved, the major benefits of the solution, the planned architecture of the solution, and a brief history of the project. Note that where this information is

Component	Description
	defined sufficiently in other documents, you might simply include a reference to those other documents if appropriate; however, it may save readers of the test plan time and effort if a limited amount of information is duplicated here; you should use your judgment. As a general rule, this section should only be about three to five paragraphs in length.
2.2 Evaluation Missions application to this Project/Phase	Provide a brief statement that defines the mission(s) for the test and evaluation effort over the scope of the plan. The governing mission statement(s) might incorporate one or more concerns including: • Find as many bugs as possible; • Find important problems, assess perceived quality risks; • Advise about perceived project risks; • Certify to a standard; • Verify a specification (requirements, design or claims); • Advise about product quality, satisfy stakeholders; • Advise about testing; • Fulfill process mandates and so forth; and • Each mission provides a different context to the test effort and changes the way in which testing should be approached.
2.3 Sources of Test Motivators	Provide an outline of the key sources from which the testing effort in this Project/ Phase will be motivated. Testing will be motivated by many things, quality risks, technical risks, project risks, use cases, functional requirements, non-functional requirements, design elements, suspected failures or faults, change requests, and so forth.
Section 3. Target Test Items	Provide a high level list of the major target test items. This list should include both items produced directly by the project development team, and items that those products rely on; for example, basic processor hardware, peripheral devices, operating systems, third-party products or components, and so forth. This may simply be a list of the categories or target areas.
Section 4. Overview of Planned Tests	This section provides a high-level overview of the testing that will be performed. This section will address all phases of testing including Unit, Integration, System, and User Acceptance phase of testing. The outline in this section represents a high level overview of both the tests that will be performed and those that will not.
4.1 Overview of Test Inclusions	Provide a high-level overview of the major testing planned for project/ phase. Note what will be included in the plan and record what will explicitly not be included in the following section titled Overview of Test Exclusions.
4.2 Overview of Other Candidates for Potential Inclusion	Give a separate overview of areas you suspect might be useful to investigate and evaluate, but that have not been sufficiently researched to know if they are important to pursue.

Component	Description
4.3 Overview of Test Exclusions	Provide a high-level overview of the potential tests that might have been conducted but that have been explicitly excluded from this plan. If a type of test will not be implemented and executed, indicate this in a sentence stating the test will not be implemented or executed and stating the justification, such as: "These tests do not help achieve the evaluation mission." "There are insufficient resources to conduct these tests." "These tests are unnecessary due to the testing conducted by xxxx." As a heuristic, if you think it would be reasonable for one of your audience members to expect a certain aspect of testing to be included that you will not or cannot address, you should note its exclusion.
Section 5. Test Approach	The Test Approach presents an overview of the recommended strategy for analyzing, designing, implementing and executing the required tests as well as managing the defects identified during testing. The test approach must also address security, Section 508 compliance, etc. Sections 3 and 4 identify what items will be tested and what types of tests will be performed. This section describes how the tests will be realized and communicated (e.g., how to handle testing with other entities, meeting frequency to discuss defects, etc.). As you identify each aspect of the approach, you should update Section 10, Responsibilities, Staffing, and Training Needs, to document the test environment configuration and other resources that will be needed to implement each aspect.
5.1 Measuring the Extent of Testing	Describe what strategy you will use for measuring the progress of the testing effort. When deciding on a measurement strategy, it is important to consider the following advice from Cem Kaner, 2000 "Bug count metrics reflect only a small part of the work and progress of the testing group. Many alternatives look more closely at what has to be done and what has been done. These will often be more useful and less prone to side effects than bug count metrics." A good measurement strategy will report on multiple dimensions including coverage (against the product and/or against the plan), effort, results, obstacles, risks (in product quality and/or testing quality), and historical trend (across iterations and/or across projects).
5.2 Identifying and Justifying Tests	Describe how tests will be identified and considered for inclusion in the scope of the test effort covered by this strategy. Describe the process that will be used to determine test suites/cases/procedures and test scripts. Provide a listing of resources that will be used to stimulate/ drive the identification and selection of specific tests to be conducted, such as Use Cases, Requirements documents, User documentation and/ or Other Reference Sources.
5.3 Conducting Tests	One of the main aspects of the test approach is an explanation of how the testing will be conducted, covering the selection of quality-risk areas or test types that will be addressed and the associated techniques

Component	Description
	that will be used. If you are maintaining a separate test strategy artifact that covers this, simply list the test types or quality-risk areas that will be addressed by the plan, and refer to the test strategy artifact for the details. If there is no separate test strategy artifact, you must provide an outline here of how testing will be conducted for each technique: how design, implementation, and execution of the tests will be done, and the criterion for knowing that the technique is both useful and successful. For each technique, provide a description of the technique and define why it is an important part of the test approach by briefly outlining how it helps achieve the Project and Evaluation Missions.
Section 6. Entry and Exit Criteria	
6.1 Project/Phase Test Plan	
6.1.1 Test Plan Entry Criteria	Specify the criteria (i.e., in addition to the Test Readiness Review) that will be used to determine whether the execution of the Test plan can begin.
6.1.2 Test Plan Exit Criteria	Specify the criteria that will be used to determine whether the execution of the Test plan is complete, or that continued execution provides no further benefit.
6.1.3 Suspension and Resumption Criteria	Specify the criteria that will be used to determine whether testing should be prematurely suspended or ended before the plan has been completely executed, and under what criteria testing can be resumed.
Section 7. Deliverables	In this section, list the various artifacts that will be created by the test effort that are useful deliverables to the various stakeholders of the test effort. Don't list all work products; only list those that give direct, tangible benefit to a stakeholder and those by which you want the success of the test effort to be measured. The Master Test Plan is required for all projects. For large scale projects the following artifacts are required in addition to this Test Plan: • System Test plan and System Test Summary Report • User Acceptance Test (UAT) Test Plan and User Acceptance Test Summary Report • User Acceptance Test (UAT) Support Plan • Data Conversion Balancing Report • Post Implementation Verification Plan • Defect Reports and Test Status Reports
7.1 Test Evaluation Summaries	Provide a brief outline of both the form and content of the test evaluation summaries, and indicate how frequently they will be produced.
7.2 Reporting on Test Coverage	Provide a brief outline of both the form and content of the reports used to measure the extent of testing, and indicate how frequently they will be produced. Give an indication as to the method and tools used to

Component	Description
	record, measure, and report on the extent of testing.
7.3 Perceived Quality Reports	Provide a brief outline of both the form and content of the reports used to measure the perceived quality of the product, and indicate how frequently they will be produced. Give an indication about the method and tools used to record, measure, and report on the perceived product quality. You might include some analysis of Incidents and Change Request over Test Coverage.
7.4 Incident Logs and Change Requests	Provide a brief outline of both the method and tools used to record, track, and manage test incidents, associated change requests, and their status.
7.5 Smoke Test Assets	Provide a brief outline of the test assets that will be delivered to allow ongoing regression testing of subsequent product builds to help detect regressions in the product quality.
7.6 Additional Work Products	In this section, identify the work products that are supplemental documentation that will lead to the successful execution of the test effort.
7.6.1 Detailed Test Results	Identify how test results will be delivered to Federal Student Aid. Either a collection of Microsoft Excel spreadsheets listing the results determined for each test suite/case, or the repository of both test logs and determined results maintained by a specialized test product is acceptable.
7.6.2 Test Guidelines	Test Guidelines cover a broad set of categories, including Good Practice Guidance, Test patterns, Fault and Failure Models, Automation Design Standards, and so forth.
Section 8. Testing Workflow	Provide an outline of the workflow to be followed by the Test team in the execution of this Test Plan. More specific details of the individual testing tasks are defined in a number of different ways, depending on project culture. For example: Defined as a list of tasks in this section of the Test Plan, or in an accompanying appendix Defined in a central project schedule (often in a scheduling tool such as Microsoft Project). Documented in individual, "dynamic" to-do lists for each team member, which are usually too detailed to be placed in the Test Plan documented on a centrally located whiteboard and updated dynamically. In this section, you should provide some descriptive text explaining the process your team uses to handle detailed task planning and provide a reference to where the details are stored, if appropriate. For Test plans, we recommend avoiding detailed task planning, which is often an unproductive effort if done as a front-loaded activity at the beginning of the project. A Test plan should describe the phases, and give an indication of what types of testing are generally planned.

Component	Description
	Note: Where process and detailed planning information is recorded centrally and separately from this Master Test Plan, you will have to manage the issues that will arise from having duplicate copies of the same information. To avoid team members referencing out-of-date information, we suggest that in this situation you place the minimum amount of process and planning information within the detailed test plan to make ongoing maintenance easier and simply reference the "Master" source material.
Section 9. Environmental Needs	This section presents the non-human resources required for the Test plan. This section shall describe the test environment and test lab facilities required to perform the testing.
9.1 Base System Hardware	This section contains a list of the hardware elements required in the test environment. The specific elements of the test system may not be fully understood in early iterations, so expect this section to be completed over time.
9.2 Base Software Elements in the Test Environment	This section contains a list of the software elements required in the test environment. The specific elements of the test system may not be fully understood in early iterations, so expect this section to be completed over time.
9.3 Productivity and Support Tools	Include the specific tools used to support the test effort.
9.4 Test Environment Configurations	This information may not be available during the definition phase of the lifecycle. Therefore, this information must be detailed in the individual Test Phase test plans.
Section 10. Responsibilities, Staffing, and Training Needs	This section presents the required resources to address the test effort outlined in the Test plan, the main responsibilities and the knowledge or skill sets required of those resources.
10.1 People and Roles	Add or delete items as appropriate.
10.2 Staffing and Training Needs	This section describes the training required to enable the testing team to effectively test the system. Training topics may include the use of testing tools, testing procedures and techniques, or other related information. Give thought to your training needs, and plan to schedule this based on a Just-In-Time (JIT) approach. There is often a temptation to attend training too far in advance of its usage when the test team has apparent slack. Doing this introduces the risk of the training being forgotten by the time it's needed. Look for opportunities to combine the purchase of productivity tools with training on those tools, and arrange with the vendor to delay delivery of the training until just before you need it. If you have enough

Component	Description
	headcount, consider having training delivered in a customized manner for you, possibly at your own site. The test team often requires the support and skills of other team members not directly part of the test team. Make sure you arrange in your plan for appropriate availability of System Administrators, Database Administrators, and Developers who are required to enable the test effort.
Section 11. Key Project/Phase Milestones	Identify the key schedule milestones that set the context for the Testing effort. Avoid repeating too much detail that is documented elsewhere in plans that address the entire project. Key Milestones will be identified for all levels of testing per phase – Unit, Integration, System and User Acceptance. The required components are included in the template for the schedule of key milestones. Also included are blank rows for additional components.
Section 12. Master Plan Risks, Dependencies, Assumptions, and Constraints	<u>Table 12-1: Risks</u> List any risks that may affect the successful execution of this Test plan, and identify mitigation and contingency strategies for each risk. Also indicate a relative ranking for both the likelihood of occurrence and the impact if the risk is realized. <u>Table 12-2: Dependencies</u> List any dependencies identified during the development of this Test plan that may affect its successful execution if those dependencies are not honored. Typically, these dependencies relate to activities on the critical path that are prerequisites or post-requisites to one or more preceding (or subsequent) activities. You should consider responsibilities you are relying on other teams or staff members external to the test effort to complete, timing and dependencies of other planned tasks, and the reliance on certain work products being produced. <u>Table 12-3: Assumptions</u> List any assumptions made during the development of this Test plan that may affect its successful execution if those assumptions are proved incorrect. Assumptions might relate to work you assume other teams are doing, expectations that certain aspects of the product or environment are stable, and so forth. <u>Table 12-4: Constraints</u> List any constraints placed on the test effort that have had a negative effect on the way in which this Test plan has been approached.
Section 13. Management Process and Procedures	Outline what processes and procedures are to be used when issues arise with the Test plan and its enactment.
13.1 Measuring and Assessing the Extent of	Define any management and procedural aspects of the measurement and assessment strategy outlined in Section 5.1, Measuring the Extent of

Component	**Description**
Testing	Testing.
13.2 Assessing the Deliverables of this Test plan	Outline the assessment process for reviewing and accepting the deliverables of this Test plan.
13.3 Problem Reporting, Escalation, and Issue Resolution	Define how process problems will be reported and escalated, and the process to be followed to achieve resolution.
13.4 Managing Test Cycles	Outline the management control process for a test cycle.
13.5 Traceability Strategies	Consider appropriate traceability strategies for: Coverage of Testing against Specifications enables measurement of the extent of testing Motivations for Testing enables assessment of relevance of tests to help determine whether to maintain or retire tests Software Design Elements enables tracking of subsequent design changes that would necessitate rerunning tests or retiring them Resulting Change Requests enables the tests that discovered the need for the change to be identified and rerun to verify the change request has been completed successfully
13.6 Approval and Signoff	Outline the approval process and list the job titles (and names of current incumbents) that initially must approve the plan, and sign off on the plan's satisfactory execution. List those that have sign-off responsibility (required); those that are accountable; those that were consulted and those that were informed.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions* contained in the beginning of Appendix E. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all of the glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Click For Document ➔	Master Test Plan.doc

Template 2, Test Readiness Review Report

Instructions

Component	Description
All templates must follow the formatting guidelines set forth in the *Template Formatting Instructions* contained in the beginning of Appendix E.	
Cover	A sample of the cover sheet is provided in the template.
Test Readiness Review Form	Complete form.
Document Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
1. Creation of Test Readiness Review Checklist	This section contains the test readiness review checklist. The list can be tailored based on the test phase requirements. Once complete, all (Integration, System and User Acceptance Tests) Test Readiness Review Checklists, meeting information and review results for a project must be included in this Test Readiness Review Report.
2. Test Readiness Review Meeting	This section contains details about the test readiness review meeting. This also includes the meeting date, test phase for which the readiness review meeting is going to be held, the list of participants and their roles.
3. Test Readiness Review Results	This section provides the Federal Student Aid decision on whether or not an application can move to the next test phase or into production. The rationale for the decision is to be provided below.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions* contained in the beginning of Appendix E. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all of the glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Click For Document ➔	Test Readiness Review Report.doc

Template 3, Phase Level Test Plan(s)

Instructions

Component	Description
All templates must follow the formatting guidelines set forth in the *Template Formatting Instructions* contained in the beginning of Appendix E.	
The guidance in this template is what is required. The creator of this test plan may add other details as needed based on the type of application under test and the requirements that are being tested.	
Cover Sheet	A sample of the cover sheet is provided in the template.
Table of Contents	A sample of the table of contents is provided in the template.
	Provide phase and system name, Contractor Project Manager signature and date, requirements status, comments if applicable, Federal Student Aid Test Manager or Project Manager signature and date, requirements status, and comments if applicable.
Document Version Control	Provide document information, document history, and release authorization.
1. Overview	This section provides a brief synopsis of the background of the project under test. State the purpose of the [Phase] test plan and state the sub phases that would be covered in this test plan. This template can be used to create a Unit Test plan, an Integration Test Plan, and a System Test Plan.
2. Planning	
2.1. Schedule	This section contains the schedule information. Include milestone dates. This information could be in the form of a work breakdown structure.
2.2. Environmental Needs	This section contains the hardware details, environment location, and test facilities that support the software test effort.
2.2.1. Test Environment	This section describes the test environment. For example – Windows 2000, Websphere, etc.
2.2.2. Testing Tools	This section contains details about the testing tools that will be used and how they will be used.
2.2.3. References	This section identifies the reference material that will be used or has been used in order to support the test effort.
2.3. Roles and Responsibilities	This section identifies the roles and responsibilities of Federal Student Aid and any Contractor for [phase] testing.
2.4. Staffing and Training Needs	This section consists of the following items: • Number of resources and skills required for the testing project • Identification of areas where training is required and description of the plan for training testers (if applicable, include

Component	Description
	the training of Federal Student Aid testers) • Include a timeframe for training
2.5. Test Team and Development Team Coordination	This section details the development team responsibilities during [phase] testing. This may include analysis of defects, creation of impact analysis statements, attending System Change Control Board meetings, etc.
3. Test Configurations	
3.1. Features to be tested	This section describes the list of features (Functions or Requirements), which will be tested or demonstrated by the test. The description is provided at a high level without going into the technical details.
3.2. Features not to be tested	This section describes the list of features (Functions or Requirements), which will not be tested.
3.3. Components to be tested	This section identifies and lists the components to be tested within the scope of this test plan.
4. Test Approach	This section includes details of testing covered by the plan and the objectives, scope, rationale and communication plan.
4.1. Test Strategy	This section describes the test strategy, which includes the following: • Processes • Assumptions • Work Products • Test data creation and test data results verification process • Test suite writing process • Peer Review Process • Regression Cycles • Configuration Management Process • Handling of meetings and other Organizational Processes • Intersystem Testing needs • Add other information, as needed
4.1.1 Test Process	This section provides a brief synopsis of the overall test process. Provide details of testing covered by the plan and the objectives, scope, and rationale.
4.1.2. Assumptions and Constraints	This section identifies the known assumptions and constraints related to the test strategy. If needed, list mitigation strategies.
4.1.3. Test Work Products	This section identifies and lists the documents that will be created to support the test effort. State the purpose of each document that will be created.

Component	Description
	List names or teams expected to review each document. This section also includes information on how the test results will be delivered to Federal Student Aid for approval purposes.
4.1.3.1. Test Artifact Details	This section describes in detail how each document will add value to the test effort; the process for delivering each document to Federal Student Aid; the Configuration Management process for documentation; peer review process of documentation; test reports; test metrics; how test will be conducted; how test suites will be created and how test results will be presented to Federal Student Aid. Other information may be added to this section.
4.1.4 Traceability	This section covers how test documentation traces back to the requirements that are under test.
4.1.5. Test Data	This section covers how test data will be created and identified for the test effort.
4.1.6. Regression Strategy	This section covers how regression testing will be performed and the regression cycles.
4.2. [xxx] System Approach and Techniques	An example would be Mainframe. This section covers exceptions pertaining to a particular subsystem of an application. If the exceptions are significant, a separate test plan must be created for the subsystem. The Contractor and the Federal Student Aid Test Manager and Project Manager will make this decision at the beginning of the project.
4.3.[xxx] System Approach and Techniques	(An example would be WEB) This section covers exceptions pertaining to a particular subsystem of an application. If the exceptions are significant a separate test plan must be created for the subsystem. The Contractor and the Federal Student Aid Test Manager and Project Manager will make this decision at the beginning of the project.
4.4. Test Estimation	This section describes the test estimation technique that will be followed for this test effort. The details should cover estimation for the test effort and how estimates will be determined related to regressing defects.
4.5. Test Readiness Review	This section contains a test readiness review process. Include a reference to the Test Readiness Review checklist, which must be an appendix to this [phase] test plan.
4.6. Pass/Fail Criteria	This section contains the conditions of the pass/fail criteria.

Component	Description
4.7. Entry and Exit Criteria	This section contains the criteria that determine when testing can begin and when the testing process can be stopped.
4.7.1. Service Level Agreement	This section covers the service level agreement (SLA) for the application under test. Connect the SLA information to exit criteria. Example: If testing a voice response system, how many responses must be recognized before Federal Student Aid will accept the product.
5. Defect Management	This section describes the defect management process that will be used for this project. Include information on documenting issues, defect classification and the change control process. Include the name of the defect management tool. Include how Federal Student Aid will be involved in the defect management process.
5.1. Documenting Issues	This section describes the process for documenting defects. This section must include when an issue is added to the defect tracking tool, meetings that must be held to discuss issues, and making the determination of when an issue becomes a defect.
5.2. Defect Classification	This section describes the different states used for classifying defects. Describe in detail the severity, complexity, status and priority classification.
5.3. Change Request	This section describes the process for handling change request. If this process is detailed in another project document, state the name of the document and provide an abbreviated version of the process.
5.4. Suspension Criteria and Resumption Requirements	This section lists the known conditions that may result in the suspension of testing. Include the process for determining when testing can resume. For example, if an application is under test and the login feature does not work, testing shall be suspended and shall resume only when the login feature has been corrected and a tester has approved that the login feature meets requirements.
6. Software Risk	This section describes known risks and mitigation strategies for each risk.
7. Lessons Learned	This section describes the lessons learned from this phase of testing. If this has already been covered in the Master Test Plan and there are no deviations, reference to the Master Test Plan is satisfactory.
8. Approvals	This section describes the approval process and includes the job titles and names of current incumbents that must approve the plan. List those that have sign-off responsibility (required), those that are accountable, those that were consulted and those that were informed.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions* contained in the beginning of Appendix E. In addition, all appendix formats must follow the naming convention identified below:

Component	Description
	• *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all of the glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Click For Document ➔	Phase Level Test Plan.doc

Template 4, Test Suites

Instructions

Component	Description
All templates must follow the formatting guidelines set forth in the *Template Formatting Instructions* contained in the beginning of Appendix E.	
Cover Sheet	A sample of the cover sheet is provided in the template.
Test Suite Form	Complete form.
Document Version Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
1. Overview	State the approach taken in order to develop the test suite template. Federal Student Aid recommends that the test suites contain test scenarios, the test scenarios contain test procedures, and the test procedures contain the test scripts.
2. Test Suite Template	Federal Student Aid recommends the use of Rational Test Manager in order to maintain a test suite template. If some other tools are used, the test suite template must contain the following fields:
Name of the Project:	Enter the name of the project.
Phase of Testing:	Enter the phase of testing for which the test suite is being executed.
Sub Phase of Testing:	Enter the sub phase of testing for which the test suite is being executed.
Test Suite:	Enter the name of the test suite.
Test Suite Status:	Enter the status of the test suite. The test suite status could be “No Run,” “In Progress,” “Passed,” “Failed” or “Blocked.”
Creation Date:	Enter the date on which the test suite was created.
Designer:	Enter the name of the tester who designed the test suite.
Estimated Development Time:	Enter the time required to design and execute the test suite.
Last Modified Date:	Enter the date on which the test suite was last modified.
Version:	Enter the version of the test suite.
Attachment:	Add attachment (If any).
Requirement Coverage:	Enter the requirement id for which the test suite is being developed..
Linked Defect:	Enter the defect id if the test suite fails. The defect id is obtained from a defect-tracking tool when a defect is entered following the failure of a test suite.

Component	Description
Additional Comments:	Enter additional comments (If any).
Test Scenario:	Enter the name of the test scenario.
Test suite:	Enter the name of the test suite.
Description:	Enter a brief description of the test suite.
Precondition:	Enter the preconditions that need to be met before executing the test suite.
Test Procedure:	Enter the sequence of action in the form of steps in order to execute the test suite.
Test suite:	Enter the test scripts that would be used to execute the test suite.
Expected Result:	Enter the results expected when the test suite is executed.
Actual Result:	Enter the actual results after the test suite is executed.
Status:	Enter the status of the test suite – The status could be "No Run," "In Progress," "Passed," "Failed" or "Blocked".
Responsible Tester:	Enter the name of the tester responsible for test execution.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions* contained in the beginning of Appendix E. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all of the glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Click For Document ➔	Test Suites.doc

Template 5, User Acceptance Test Plan

Instructions

Component	Description
All templates must follow the formatting guidelines set forth in the *Template Formatting Instructions* contained in the beginning of Appendix E.	
Cover	A sample of the cover sheet is provided in the template.
User Acceptance Test Plan Form	Complete form.
Document Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
1. Overview	This section provides a brief synopsis of the background of the project under test. State the purpose of the User Acceptance Test Plan, and state the sub phases (Functional testing and/or Intersystem Testing and/or Regression Testing) that will be covered in this test plan.
2. Planning	
2.1 Schedule	This section contains the schedule information. Include milestone dates. This information could be in the form of a work breakdown structure.
2.2 Environmental Needs	This section contains the hardware details, environment location, and test facilities that support the software test effort and any technical need that Federal Student Aid may request from a contractor.
2.2.1 Test Environment	
2.2.2 Testing Tools	This section contains details about the testing tools that will be used and how they will be used.
2.2.3 References	This section identifies the reference material that will be used or has been used in order to support the test effort.
2.3 Roles and Responsibilities	This section identifies the roles and responsibilities of Federal Student Aid and Contractor for system testing. Identify individuals by name, role and list the timeframe of their participation in the User Acceptance test effort.
2.4 Staffing and Training Needs	This section consists of the following items: • Number of resources and skills required for the testing project • Identification of areas where training is required and a plan for training testers • Include a timeframe for training
3. Requirements Phase	This section describes the Federal Student Aid test team's involvement in the requirements phase of the project. This will include checking

Component	Description
	testability of requirements, review of functional specifications, and detail design and analysis of the requirements tracking matrix (RTM).
4. Review of Contractor Results	This section describes the Federal Student Aid test team's process for reviewing the contractor's System Test results.
5. Test Configurations	
5.1 Features to be Tested	This section describes the list of features (Functions or Requirements), which will be tested or demonstrated by the test. The description is provided at a high level without going into the technical details.
5.2 Features Not to be Tested	This section describes the list of features (Functions or Requirements), that will not be tested.
5.3 Components to be Tested	This section identifies and lists the components to be tested within the scope of this test plan.
6. Test Approach	This section includes details of testing covered by the plan and the objectives, scope, rationale and communication plan.
6.1 Test Strategy	This section includes the test strategy.
6.1.1 Test Process	This section provides a brief synopsis of what it takes to test. Provide details of testing covered by the plan and the objectives, scope, and rationale.
6.1.2 Assumptions and Constraints	This section identifies the known assumptions and constraints that feed into the test strategy. If needed, list mitigation strategies.
6.1.3 Test Work Products	This section identifies and lists the documents that will be created to support the test effort. State the purpose of each document, list names or teams, create and review the document.
6.1.3.1 Test Artifact Details	This section identifies the process for creating test suites. If the contractor is to provide the test suites to Federal Student Aid, then this document must indicate this and include the process for providing the contractor information for developing the test suites, the approval process and the review process of the test suites.
6.1.4 Regression Strategy	This section covers how regression testing will be performed and the regression cycles.
6.2 Traceability	This section describes how test documentation will trace back to the requirements that are under test.
6.3 Test Data	This section covers how test data will be created and identified for the test effort.
6.4 Test Estimation	This section will describe the test estimation technique that will be followed for this test effort. The details should cover estimation for the test effort and how estimates will be determined related to regressing defects.

Component	Description
6.5 Test Readiness Review	This section contains a test readiness review process. Include a reference to the Test Readiness Review checklist, which must be an appendix to this User Acceptance Test Plan.
6.6 Pass/Fail Criteria	This section contains the conditions of the pass/fail criteria.
6.7 Entry and Exit Criteria	This section contains the criteria that determine when testing can begin and when the testing process can be stopped.
6.8 Xxx System Approach and Techniques	This section covers minor exceptions pertaining to a particular subsystem of an application. If the exceptions are significant a separate test plan must be created for the subsystem. The contractor and the Federal Student Aid Test Manager and Project Manager will make this decision in the beginning phases of the project.
7. Defect Management	This section describes the defect management process that will be used for this project. Include information on documenting issues, defect classification and change control process. Include the name of the defect management tool. Include how the contractor will be involved in the defect management process.
7.1 Documenting Issues	This section describes the process for documenting defects. This must describe when an issue is added to the defect tracking tool, meetings that must be held to discuss issues and making the determination of when an issue becomes a defect.
7.2 Defect Classification	This section describes the different states used for classifying defects. Describe in detail the severity, complexity, status and priority classification.
7.3 Change Request	This section describes the process for handling change requests. If this process is detailed in another project document, state the name of the document and provide an abbreviated version of the process.
7.4 Suspension Criteria and Resumption Requirements	This section lists the known conditions that may result in the suspension of testing a function of the system under test. Include the process for determining the resumption of the test. For example, if an application is under test and the login feature does not work, testing activity shall be suspended and shall resume only when the login feature has been corrected and a tester has approved that the login feature meets expectations.
8. Software Risk	
8.1 Risk and Mitigation Strategy	This section details known risk and mitigation strategies for each risk.
9. Lessons Learned	This section details the lessons learned process for this phase of testing.
10. Approvals	This section describes the approval process and lists the job titles (and names of current incumbents) that must approve the plan. List those that

Component	Description
	have sign-off responsibility (required), those that are accountable, those that were consulted, and those that were informed.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions* contained in the beginning of Appendix E. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all of the glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Click For Document ➔	User Acceptance Test Plan.doc

Template 6, User Acceptance Test Support Plan

Instructions

Component	Description
All templates must follow the formatting guidelines set forth in the *Template Formatting Instructions* contained in the beginning of Appendix E.	
Cover	A sample of the cover sheet is provided in the template.
User Acceptance Test Support Plan Form	Complete form.
Document Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
1. Overview	This section will give a brief synopsis of the background of the project under test. State the purpose of the User Acceptance Test Support Plan and state the sub phases (Functional testing and/or Intersystem Testing and/or Regression Testing) that would be covered in this test plan.
2. Schedule	This section will include the timeline of the test. It is acceptable to have a link to the Work Breakdown Structure.
3. Test Environment Requirements	This section will list the hardware and work space required and that will be provided by the contractor during User Acceptance Testing.
4. Responsibilities	This section will list the responsibilities of the contractor during User Acceptance Testing. This section will also include responsibilities that Federal Student Aid has to the contractor during User Acceptance Testing.
4.1 Setup	This section will list the tasks that the contractor will perform to prepare the User Acceptance Testing environment. This may include creation of test suites, tracking tools, etc.
4.2 Execution Support	This section will list the support that the contractor will provide during User Acceptance Testing. This may include defect resolution, meeting attendance, etc.
5. Staffing and Training Needs	This section will list the training that the contractor will provide to Federal Student Aid.
6. Test Configurations	
6.1 Features to be tested	This section describes the list of features (Functions or Requirements), which will be tested or demonstrated by the test. The description is given at a high level without going into the technical details. State that these are the features that Federal Student Aid will approve.
6.2 Features not to be tested	This section describes the list of features (Functions or Requirements) which will be tested or demonstrated by the test. The description is given at a high level without going into the technical details. State that

Component	Description
	these are the features that Federal Student Aid will not test during User Acceptance Testing.
7. Test Approach	This section will include the approach that will be used. For example, state where the testing will take place, and the access that Federal Student Aid will have to the contractor that is supporting the test effort. State level of participation in the User Acceptance Testing Test Readiness Review process. State the services that the contractor will provide to Federal Student Aid during User Acceptance Testing.
7.1 Pass/Fail Criteria	This section contains the conditions of determining pass/fail criteria during User Acceptance Testing.
8. Suspension Criteria and Resumption Requirements	This section will list the support the contractor will provide to Federal Student Aid in cases where User Acceptance Testing is suspended due to unforeseen suspension of the test process.
9. Test Work Products	This section will list the work products that will be created by the contractor during User Acceptance Testing.
10. Tasks	This section will list the tasks that the contractor will perform during User Acceptance Testing.
11. Data Request	This section will include how data request will be delivered to Federal Student Aid for User Acceptance Testing. Include a data request template in the appendix of this document and provide this template to Federal Student Aid as early as possible and no later than during the beginning phases of System Testing.
12. Software Risk	This section will detail known risk (under contractor control) that may occur during User Acceptance Testing and mitigation strategies for each risk.
13. Test Readiness Criteria	This section will include Test Readiness Review information such as what the TRR will ensure prior to User Acceptance Testing.
14. Test Completion Criteria	This section will include information on how the contractor and Federal Student Aid determine when User Acceptance Testing is complete.
15. Approvals	This section will outline the approval process and list the job titles (and names of current incumbents) that initially must approve the plan, and sign off on the plan's satisfactory execution. List those that have sign-off responsibility (required), those that are accountable, those that were consulted and those that were informed.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions* contained in the beginning of Appendix E. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all of the*

Component	Description
	glossary terms used in the document. • *Appendix C will start all subsequent appendices.*
Click For Document ➔	User Acceptance Test Support Plan.do

Template 7, Post Implementation Verification Test Plan

Instructions

Component	Description
All templates must follow the formatting guidelines set forth in the *Template Formatting Instructions* contained in the beginning of Appendix E.	
Cover	A sample of the cover sheet is provided in the template.
Post Implementation Verification Form	Complete form.
Document Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
1. Overview	This section will give a brief synopsis of the background of the project under test. State the purpose of the Post Implementation Verification and state the sub phases (Functional testing and/or Intersystem Testing and/or Regression Testing) that would be covered in this test plan.
2. Planning	
2.1 Schedule	This section contains the schedule information. Include milestone dates. This information could be in the form of a work breakdown structure.
2.2 Roles and Responsibilities	This section will identify the roles and responsibilities of Federal Student Aid and Contractor for system testing.
3. Critical Functional Areas to Be Tested	This section lists the critical features (Functions or Requirements), which will be analyzed during their first cycle in the production environment.
4. Verification of Critical Features	This section describes how each critical feature will be verified and how the feature will be deemed successful.
5. Reporting Results to Federal Student Aid	This section will include details of how test results will be communicated to Federal Student Aid. Must include how major failures will be communicated to Federal Student Aid.
6. Post Implementation Checklist	This section will include a checklist for each item under test. The information may be included in an EXCEL spreadsheet. At a minimum the fields included must be: • Verification Item • Verification Method • Verification Timeframe • Review Date • Expected Result • Actual Result

Component	Description
	• Pass/Fail • Resource that will be verifying item • Comments • Supporting Information • Contractor Sign-off • Federal Student Aid Sign-off
7. Approvals	This section will outline the approval process and list the job titles (and names of current incumbents) that initially must approve the plan, and sign off on the plan's satisfactory execution. List those that have sign-off responsibility (required), those that are accountable, those that were consulted and those that were informed.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions* contained in the beginning of Appendix E. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all of the glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Click For Document ➔	Post Implementation Test Plan.doc

Template 8, Test Summary Report

Instructions

Component	Description
All templates must follow the formatting guidelines set forth in the *Template Formatting Instructions* contained in the beginning of Appendix E.	
Cover Sheet	A sample of the cover sheet is provided in the template.
Document Version Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Test Summary Report Form	Complete form.
Table of Contents	A sample of the table of contents is provided in the template.
Executive Summary	This section provides a brief summary of the project, critical requirements and the findings of the test effort. This summary must be no longer than 1½ pages.
1.Purpose of the Test Summary	This section describes the purpose of this document.
2. Test Items	This section contains a high-level description of the items under test.
2.1 Reference Material	This section contains the documents that were used for the test effort. Such documentation includes requirements, test suites, test results, data, etc. Provide the document name, state if the document is available, and describe how to obtain access to it.
2.1.1 Environment	This section contains details about the environment that was used for the test effort.
3. Variances	This section describes the variances and reasons for variances that occurred during the test effort.
3.1.1 Test Dates	This section describes variances in the baseline test start dates and end dates.
3.1.2. Outages	This section describes the outages (planned or not planned) that occurred during the test effort that may have caused a delay or workarounds that were not planned.
3.1.3 Others	This section describes each type of variance.
4. Comprehensive Assessment	This section includes an evaluation of the testing and test process in terms of the documented test objectives.
5. Summary of Results	This section describes the overall status of defects. It includes the total number of defects, severity, priority, cost, defect patterns, and any open or unresolved defects. The focus should include positive and negative trends that occurred during the test.
6. Evaluation	This section provides a total evaluation of the test effort. This section should assess the quality of the software. Include both positive and

Component	Description
	negative results.
7. Summary of Activities	This section includes the planned activities and the changes to the plans. Include reasons for deviations and the impact on the test effort.
8. Approvals	This section lists those responsible for reviewing and approving this document. The approvers must be the same ones that initially approved the test plan associated with the test summary report.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions* contained in the beginning of Appendix E. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all of the glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Click For Document ➔	Test Summary Report.doc

Template 9, Performance Test Plan

Instructions

Component	Description
All templates must follow the formatting guidelines set forth in the *Template Formatting Instructions* contained in the beginning of Appendix E.	
Note: *All sections outlined below are required. However, additional sections may be included based on the type of system being tested.*	
Cover Sheet	A sample of the cover sheet is provided in the template.
Document Version Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
Section 1. Overview	High-level summary of the application.
Section 2. Introduction	High level view of the application and testing approach.
Section 3. Test Items	The items of software, hardware, and combinations of these that will be tested.
Section 4. Features to be Tested	A list of which functions/features of the application will be tested.
Section 5. Features Not to be Tested	A description of which functions/features of the application will not be tested.
Section 6. Approach	The process used for testing.
Section 7. Item Pass/Fail Criteria	Lists the explicit Pass/Fail criteria for each test run.
Section 8. Suspension Criteria and Resumption Requirements	The circumstances that would cause testing to stop and restart.
Section 9. Test Deliverables	Test documents and other deliverables which will be produced. Deliverables are: • Test Plan • Daily Test Reports (summary E-mail delivered within 24 hours following each test session) • Test Report
Section 10. Testing Tasks	List of tasks, their dependencies, the elapsed time to complete, and resources required.
Section 11. Environmental Needs	What is needed in the way of testing software, hardware, etc.
Section 12. Responsibilities	List of who is responsible for delivering the various parts of the plan.

Component	Description
Section 13. Staffing and Training Needs	The people and skills needed to deliver the plan.
Section 14. Schedule	When testing will occur.
Section 15. Risks and Contingencies	This defines all other risk events, their likelihood, impact and counter measures.
Section 16. Approvals	The signatures of the various stakeholders to show they agree in advance with the plan.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions* contained in the beginning of Appendix E. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all of the glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Click For Document ➔	Performance Test Plan.doc

Template 10, Performance Testing Test Report

Instructions

Component	Description
All templates must follow the formatting guidelines set forth in the *Template Formatting Instructions* contained in the beginning of Appendix E.	
Note: *Sections in this plan may be added or removed based on the type of system being tested.*	
Cover Sheet	A sample of the cover sheet is provided in the template.
Document Version Control	Provide information concerning the version of the document, the date of the change, and a description of the changes.
Table of Contents	A sample of the table of contents is provided in the template.
Executive Summary	High-level summary of the application.
1.Performance Test Results	This section contains a summary of the major findings during the performance testing.
1.1 Issues	
1.1.1 Open Issues	A list of the open issues.
1.1.2 Resolution	A brief description of the issues and how they were resolved.
1.2 Static Tests	
1.3 Peak Tests	
1.3.1 XXX Peak	
1.3.2 XXX Peak	
2. Capacity Planning and Configuration	
Note: *Capacity Planning is for the production system and not for the performance test environment.*	
2.1 Capacity Planning	Details of the planning (e.g., the number of servers required to support the load and the number of processes running on servers.)
2.2 Configuration	Details of system architecture of the performance test environment. If available, provide a diagram of the environment.
4. Production Performance Readiness	A determination of the overall evaluation of the readiness for the application to run in production based on the Performance Test Results. This is a recommendation provided by the Performance Test Team to the Application Owner.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions* contained in the beginning of Appendix E. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only;*

Component	Description
	list all of the acronyms used in the document. • *Appendix B is reserved for the glossary only; list all of the glossary terms used in the document.* • *Appendix C will start all subsequent appendices.*
Click For Document ➔	Performance Testing Test Report.doc

Template 11, Hardware Test Plan

Instructions

Component	Description
All templates must follow the formatting guidelines set forth in the *Template Formatting Instructions* contained in the beginning of Appendix E.	
Cover Sheet	A sample of the cover sheet is provided in the template.
Test Plan Certification	Provide information about the project approvals.
Document Version Control	Provide information concerning the version of the document, the date of the change, and a description of the change.
Table of Contents	A sample of the table of contents is provided in the template.
Section 1. Overview	This section will give a brief synopsis of the background of the project under test. State the purpose of the Hardware Test Plan.
Section 2. Planning	
2.1 Schedule	This section contains the schedule information. Include milestone dates. This information could be in the form of a work breakdown structure.
2.2 Environmental Needs	This section contains the hardware details, environment location, test facilities that support the software test effort.
2.2.1 Test Environment	
2.2.2 Testing Tools	This section contains details about the testing tools that will be used and how they will be used.
2.2.3 References	This section will identify the reference material that will be used or has been used in order to support the test effort.
2.3 Roles and Responsibilities	This section will identify the roles and responsibilities of Federal Student Aid and the Contractor for system testing.
2.4 Staffing and Training Needs	This section consists of the following items: • Number of resources and skills required for the testing project • Identification of areas where training is required and details the plan for training testers (if applicable include training of Federal Student Aid testers) • Include timeframe for training
Section 3. Test Configurations	
3.1 Required Features	This section describes the list of features, which will be tested or demonstrated by the test.
3.2 Optional Features	This section describes the list of hardware features that will not be tested. Only list those that some would naturally believe would be tested given the nature of the project.

Component	Description
Section 4. Test Approach	This section will include details of testing covered by the plan and the objectives, scope, rationale and communication plan.
4.1 Test Strategy	This section will include the test strategy, which includes the following: • Process • Assumptions • Work Products • Test data needed if any to test hardware • Types of reports and metrics • Test case writing process • Peer Review Process • Configuration Management Process • Handling of meetings and other Organizational Processes • Add other information as needed
4.1.1 Test Process	This section gives a brief synopsis of what it takes to test. Provide details of testing covered by the plan and the objectives, scope and rationale.
4.1.2 Assumptions and Constraints	This section will identify the known assumptions and constraints that feed into the test strategy. If needed list mitigation strategies.
4.1.3 Test Work Products	This section will identify and list the documents that will be created to support the test effort. State the purpose of each document that will be created. List names or team expected to review each document. **This section will also include information on how the test results will be delivered to Federal Student Aid for approval purposes.**
4.1.4 Traceability	This section covers how test documentation traces back to the hardware functions that are under test.
4.1.5 Test Data	This section covers how test data if needed will be created and identified for the test effort.
4.2 Test Estimation	This section will describe the test estimation technique that will be followed for this test effort. The details should cover estimation for the test effort and how estimates will be determined related to regressing defects.
4.3 Test Readiness Review	This section contains a test readiness review process. Include a reference to the Test Readiness Review checklist, which is found in Appendix C.
4.4 Pass/Fail Criteria	This section contains the conditions for the pass/fail criteria.
4.5 Entry and Exit Criteria	This section contains the criteria that determines when testing can begin and when the testing process can be stopped.
4.5.1 Service Level	This section covers the service level agreement for the application

Component	Description
Agreement	under test. Include tie to exit criteria. Example: If testing voice response system how many responses must be recognized before Federal Student Aid will accept the product.
Section 5. Defect Management	This section will describe the defect management process that will be used for this project. Include information on documenting issues, defect classification and change control process. Include the name of the defect management tool. **Include how Federal Student Aid will be involved in the defect management process.**
5.1 Documenting Issues	This section will describe the process for documenting defects. This must include when an issue is added to the defect tracking tool, meetings that must be held to discuss issues and making the determination of when an issue becomes a defect.
5.2 Defect Classification	This section will describe the different states used for classifying defects. Describe in detail the severity, complexity, status and priority classification.
5.3 Change Request	This section will describe the process for handling change requests. If this process is detailed in another project document state the name of the document and provide an abbreviated version of the process.
5.4 Suspension Criteria and Resumption Requirements	This section will list the known conditions that may result in the suspension of testing a function of the hardware under test. Include the process for determining the resumption of the test. For example, if the hardware under test does not work and causes all other testing to fail, testing activity will be suspended and will resume only when the hardware functionality meets expectations.
Section 6. Risk	
6.1 Risk and Mitigation Strategy	This section will detail known risk and mitigation strategies for each risk.
Section 7. Lessons Learned	This section will detail the lessons learned process for this phase of testing.
Section 8. Approvals	This section will outline the approval process and list the job titles and names of current incumbents that must approve the plan, and sign off on the plans satisfactory execution. List those that have sign-off responsibility (required), those that are accountable, those that were consulted and those that were informed.
Appendices	Instructions for completing the appendices can be found in the *Template Formatting Instructions* contained in the beginning of Appendix E. In addition, all appendix formats must follow the naming convention identified below: • *Appendix A is reserved for acronyms and abbreviations only; list all of the acronyms used in the document.* • *Appendix B is reserved for the glossary only; list all of the*

Component	Description
	glossary terms used in the document. • *Appendix C will start all subsequent appendices.*
Click For Document ➔	Hardware Test Plan.doc

Templates 12 Through 22

#	Description	Document
12.	Accessibility Review Request Form	Accessibility Review Request Form.doc
13.	Performance Test Daily Reports	Performance Test Daily Reports.xls
14.	Test Schedule and Effort Variances	Test Schedule and Effort Variances.xls
15.	Test Defect Metrics	Test Defect Metrics.xls
16.	Defect Report	Defect Report.xls
17.	Defect Severity and Current State	Defect Severity-Current Stat
18.	Test Execution Report	Test Execution Report.xls
19.	Code Coverage Report	Code Coverage Report.xls
20.	Tester Team Turnover	Tester Team Turnover.xls
21.	High Level Test Summary	High Level Test Summary.doc

#	Description	Document
22.	Incident Report By Testing Phase	Incident Report By Testing Phase.xls

Appendix F – Bibliography and References

Appendix F: Bibliography and References

- http://www.geocities.com/xtremetesting/EstimatingTestingProcess.html - Estimating the Software Testing Process - Youri Kroukov
- Operational Excellence through efficient software testing metrics - Ramesh Pusala
- Rational Unified Process - Rational Software Corporation
- Guide to the CSTE Common Body of Knowledge - Quality Assurance Institute
- IEEE 829-1998
- http://www.bullseye.com/coverage.html - Steve Cornett
- http://www.webspiders.com/en/testing_lifecycle.asp
- Testing IT: An Off-the-Shelf Software Testing Process - John Watkins
- www.research.umbc.edu\ncseaman\ifsm698\lect0910.ppt
- http://www.umbc.edu/oit/iss/syscore/wiki/Main_Page
- www.eforceglobal.com

This Handbook was created in accordance with the following Federal Student Aid policies:

- Department of Education, ACS Directive OCIO: 1-106 Lifecycle Management (LCM) Framework, dated 12/02/2005.
- Department of Education, ACS Directive OCIO: 3-105 Procuring Electronic and Information Technology (E&IT) in Conformance with Section 508 of the Rehabilitation Act of 1973 as Amended, dated May 1, 2006.
- Department of Education, ACS Directive OM: 5-101: Contractor Employee Personnel Security Screenings; dated 03/30/2005.
- Department of Education, Federal Student Aid Acquisition Process Handbook, Version 1.1, January 15, 2008
- Department of Education, Federal Student Aid Style Guide, Version 1, dated July 21, 2006.
- Department of Education, Federal Student Aid System Integration and Testing Process Handbook Volumes 1, 2 and 3, dated January 12, 2001.
- Department of Education, Federal Student Aid Virtual Data Center Configuration Management Database Data Dictionary, Version 1.1, dated November 7, 2007.